NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

ANALYSIS AND MODELING OF THE VIRTUAL HUMAN INTERFACE FOR THE MARG BODY TRACKING SYSTEM USING QUATERNIONS

by

Alper Sinav

March 2002

Thesis Advisor:	Michael Zyda
Thesis Co-Advisors:	Beny Neta
	Xiaoping Yun

This thesis done in cooperation with the MOVES Institute

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** March 2000	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE: Analysis and Modeling of the Virtual Human Interface for the MARG body Tracking System Using Quaternions			**5. FUNDING NUMBERS**
6. AUTHOR(S) Alper SINAV			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution unlimited)			**12b. DISTRIBUTION CODE**

13. ABSTRACT *(maximum 200 words)*

Mathematicians have used quaternions for about a hundred years. Today they are an important part of computer graphics and simulation systems. This thesis takes an analytical approach to quaternions by using them in the construction of a virtual human for sourceless Magnetic Accelerometer Rate Sensor (MARG) body tracking system.

Virtual citizens will be a reflection of our personalities in cyberspace. Prophecies say they may take control in the virtual world and govern themselves too. One of the objectives of this thesis is to design a seamless and realistic humanoid from laser scan data clouds. This humanoid will be compatible with motion capture systems and networked virtual environments.

Second objective of this thesis is to search for the answers related to the optimal real-time representation of an articulated virtual human, maintaining a high level of visual fidelity within networked cyberspace. While visual detail and fidelity have been and will continue to be a major ongoing interest within the computer graphics community, the idea of sourceless body tracking is still in its early stages. MARG body tracking is one of the successful approaches to body tracking systems. This thesis proposes a networked quaternion based real-time virtual human interface for the MARG body tracking system..

14. SUBJECT TERMS Virtual Reality , Motion Capture ,Body Tracking, Virtual Human			**15. NUMBER OF PAGES** 92
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

ANALYSIS AND MODELING OF THE VIRTUAL HUMAN INTERFACE FOR THE MARG BODY TRACKING SYSTEM USING QUATERNIONS

Alper Sinav
Lieutenant Junior Grade , Turkish Navy
B.S.C.E, Turkish Naval Academy, 1996

Submitted in partial fulfillment of the
requirements for the degrees of

MASTER OF SCIENCE IN COMPUTER SCIENCE
AND
MASTER OF SCIENCE IN APPLIED MATHEMATICS

from the

NAVAL POSTGRADUATE SCHOOL
March 2002

Author:
Alper Sinav

Approved by:
Michael Zyda,
Thesis Advisor

Beny Neta,
Co-Advisor

Xiaoping Yun,
Co-Advisor

Christopher Eagle
Acting Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Mathematicians have used quaternions for about a hundred years. Today they are an important part of computer graphics and simulation systems. This thesis takes an analytical approach to quaternions by using them in the construction of a virtual human for sourceless Magnetic Accelerometer Rate Sensor (MARG) body tracking system.

Virtual citizens will be a reflection of our personalities in cyberspace. Prophecies say they may take control in the virtual world and govern themselves too. One of the objectives of this thesis is to design a seamless and realistic humanoid from laser scan data clouds. This humanoid will be compatible with motion capture systems and networked virtual environments.

Second objective of this thesis is to search for the answers related to the optimal real-time representation of an articulated virtual human, maintaining a high level of visual fidelity within networked cyberspace. While visual detail and fidelity have been and will continue to be a major ongoing interest within the computer graphics community, the idea of sourceless body tracking is still in its early stages. MARG body tracking is one of the successful approaches to body tracking systems. This thesis proposes a networked quaternion based real-time virtual human interface for the MARG body tracking system.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank to Prof. Mike Zyda for letting me work with MOVES Institute, to Prof Beny Neta for giving me the opportunity of another degree in Applied Mathematics, and to Xiaoping Yun for everything he taught to me.

My wife, Esin , thank you   for your patience.

My little son, Gokalp , thank you for your smile.

And, finally, I thank to God for everything.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. MOTIVATION

Since the earliest day of the art form, animators have observed the movement of real creatures, in order to create animated motion. Sometimes this simply takes the form of an artist carefully observing nature for inspiration. Another process is to transfer the movement from recording of the movement to the animated objects. The earliest mechanism for doing this was the Rotoscope, a device that projected frames of film onto the animator's workspace, providing the animator with a guide for their drawings.

Today computer power and improvements in micro machine technology allow researchers to design special sensors in order to track the human body. The magnetic, Accelerometer, Rate sensor (MARG) based body tracking system is an ongoing project at the Naval Postgraduate School and is a good example of this technology. The MARG body tracking system does not depend on an external source for body tracking and this provides a remote body tracking capability to the system with a special position-tracking device. This thesis concentrates on using outputs of a MARG system for a virtual human.

B. PROBLEM STATEMENT

Even though there is a big increase in central processor and graphics processor units, it is still too difficult to animate full human body motion in real time. There had been a previous work to achieve this goal by using segmented human body [DUTTON2001]. The segmented human body has a joint break problem when it moves. This thesis focused on this problem and tried to answer the question of how to create a seamless skin model with segmented skeleton, which uses quaternion information from the MARG body tracking system.

C. PROPOSED SOLUTION

As stated in the problem statement this thesis looks for a method to construct a virtual human. [DUTTON2001] proposed a solution to design a virtual human from laser scans by using H-Anim 1.1 specification. This research follows the same way with a different approach. Instead of segmented skeleton structure, this thesis used a skin model with a skeleton structure, which is an application of H-Anim 2000 standard. The virtual human will have the ability to position itself by using Global Position System data and to orient its segments by using MARG sensor quaternion data

.

D. DESIGN CONSIDERATIONS AND CONSTRAINTS

There are three constraints for this research. The first constraint is related to the real-time rendering capabilities of computer graphics systems. As the level of detail for laser scans increases ,the speed of real-time rendering decreases. So this trade off between realistic appearance and real-time rendering has leaded this research to search for an optimum solution.

The second constraint is the number and current capabilities of MARG sensors. There are only three MARG sensors finished right now. This prevents experimentation with full body articulation. A simulated quaternion data broadcasting system was designed because of the lack of real-time networking for processed data.

Finally it is noted that currently there is no computer graphics system that uses quaternions for graphical transformation purposes. A quaternion to axis angle transformation class is written to simulate the concept.

E. GOALS

The objective of this thesis is to analyze the quaternion rotations in detail and to construct a virtual human interface for a quaternion based MARG motion capture system. Designing a realistic and real-time virtual human interface has three steps. First step is the construction of a realistic appearance. This is achieved by using laser scan data cloud. The second step is segmenting this static virtual human data cloud for animations by using H-Anim 2.0 specification. The final step is interfacing virtual human and MARG body tracking system through the networked virtual environment.

F. ORGANIZATION

Chapter II of this thesis provides some fundamental information about quaternions. Especially for definition of quaternions, Sir William Hamilton's method, which starts with vector algebra, is used. Also in this chapter some basic information about MARG Body Tracking System and Virtual Humans is given. Chapter III provides an overview of modeling a virtual Human from Laser Scans and its deficiencies. Chapter IV discusses the application of quaternion based MARG sensors in Virtual Environments. Chapter V is the summary and the conclusions from the experience. Because the thesis is part of an ongoing project , this chapter also proposes some future topics for research.

THIS PAGE INTENTIONALLY LEFT BLANK

II. BACKGROUND

A. QUATERNIONS

1. Elements Of Quaternion

The quotient of two vectors is called a **Quaternion.** .[HAMILTON1899]

Let α and β' (Fig.1) be two vectors drawn from O and O' respectively and not lying on the same plane; and let their quotient be designated in the usual way by α / β' .

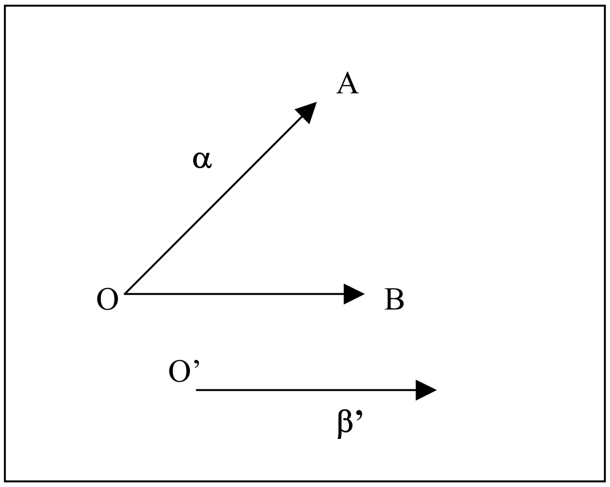


Figure 1. Quaternions as quotient of two vectors

Whatever their relative positions, we may always conceive that one of these vectors, as β', may be moved parallel to itself so that the point O' shall move over the line OO' to O. The vectors will then lie in the same plane since neither the direction nor the length of β' has been changed during this parallel motion, we have β = β'**.** And the quotient of any two vectors, α and β' will be the same as two equal co-initial vectors as α and β. We are than to determine the ratio α/β in which α and β lie in the same plane and have a common origin O.

Whatever the nature of this quotient, we are to regard it as some factor which operating on the divisor produces the dividend i.e. causes β to coincide with α in direction and length, so that if this quotient is **q**, we shall have by definition,

$$q.\beta = \alpha \qquad \text{(Eq.1)}$$

$$\text{when} \quad \frac{\alpha}{\beta} = q \qquad \text{(Eq. 2)}$$

If at the point O' we suppose a vector **O'C** = γ to be drawn, not parallel to the plane AOB, and that this vector be moved as before, so that O' falls at O, the plane which after this motion,γ will coincide on α, will differ from the plane of α and β, so that if the quotient; α/β = **q'** , **q** and **q'** will differ because their planes differs. Hence we conclude that the quotients **q** and **q'** cannot be the same if α , β and γ are not parallel to one plane, therefore that the position of the plane of α and β must enter into our conception of quotient **q**.

Again, if γ is a vector **O'C**, parallel to the plane AOB, but differing as a vector from β', then when moved, as before, into the plane AOB, it will make with α an angle other than BAO. Hence the angle between α and β must also enter into our conception of q. The direction of the angle is also, called direction of rotation another important contribution to q.

Finally, if the lengths of β and γ differ, then α/β = q will still differ from α/γ = q'. Therefore the lengths of vectors must also enter into conception of q.

We have found the quotient q, regarded as an operator, which changes β into α, depending upon the plane of vectors, the angle between them and the ratio of their lengths. Since two angles are requisite to fix a plane, it is evident that q depends on four elements and performs two distinct operations. First operation is named as stretching of β, as to make it the same length as α and the second operation is turning of β, as to cause it to coincide with α in direction.

Of the four elements of quaternion;

- Turning operation depends on three angles

  Two angles to fix the plane of rotation

  One angle to fix the amount of rotation

- Stretching depends on the ratio of the vector lengths

A quaternion is a complex quantity, which is decomposable into two factors. One of them stretches and shortens the vector divisor in order to provide the same length of vector dividend. This is called **Tensor of the quaternion**. Other part serves to turn the divisor for coincidence into dividend vector, which is called as **Versor of the quaternion** by Sir William Hamilton.[HAMILTON1899] These factors are denoted as **T(**q) and **U(**q) respectively. So q maybe written

$$q = \mathbf{T}(q) \,.\, \mathbf{U}(q). \qquad \text{(Eq.3)}$$

2. Properties of a Quaternion

a. Equality of Quaternions

By using previous considerations we can define the equality of quaternions. If the plane of α and β is moved parallel to itself or if the angle AOB (Fig 1) , remaining constant in magnitude and measured in the same direction, is rotated about an axis through o perpendicular to the plane; or the absolute lengths of α and β vary so that their ratio remains constant, q will remain same.

Simply, define

$\alpha / \beta = q$ and $\alpha' / \beta' = q'$

$q = q'$ if and only if

- The vector lengths are in the same ratio and
- The vectors are in the same or parallel planes, and

- The vectors make with each other the same angle both in magnitude and direction.

We should recall that again that q is treated as an operator that converts β into α by qβ = α.

b. Quadrantal Versors

As quaternion q operates on vector β it involves not only as rotation angle but also as the direction of this rotation angle. This leads us to the concept of a reference frame. By positive direction we assume the simple right hand rule.

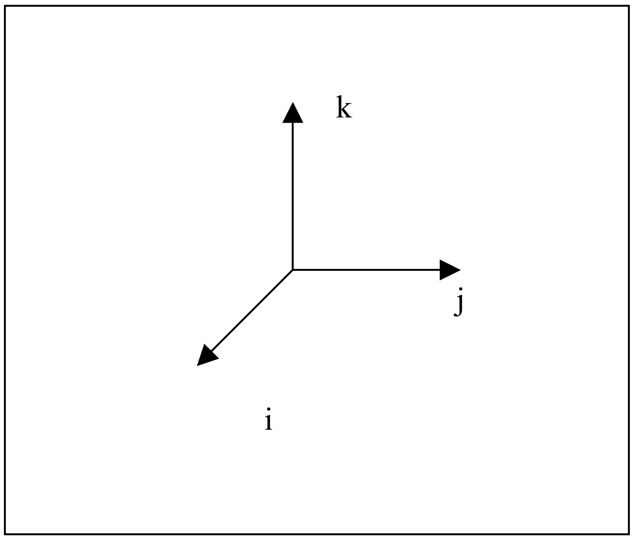


Figure 2. Quadrantal Versors

Let i*, j ,k (Fig. 2)* represent unit vectors at right angles to each other. The effect of any unit vector acting as a multiplier upon another at right angles to it has been defined to be the turning of the latter in a positive direction in a plane perpendicular to the operator or multiplier through an angle of 90°. Thus i, operating on j, produce k. This operation is called multiplication and is expressed usually;

$$ij = k. \qquad \text{(Eq.4)}$$

Another approach is a result of quotient representation;

$$k/j = i. \qquad \text{(Eq.5)}$$

It has to be pointed out that this multiplication is not that of algebra; it is a revolution. Neither k in Equation 4 is a numerical product nor i in Equation 5 is a numerical quotient. This kind of multiplication is called *geometric.*

The following table may be obtained in accordance with above.

ij = k	*k/j = i*
jk = I	*i/k = j*
ki = j	*j/i = k*
Ji = -k	*-k/i = j*
kj = -I	*-i/j = k*
Ik =-j	*-j/k =i*

Table 1. Products of Quadrinomial Versors

Because the effect of i, j, k, is a rotation from one direction to another with an angle of 90° they are called *quadrantal versors*

For a long time Sir William Hamilton was not able to find an appropriate way to multiply quaternions. While walking along a canal in Dublin he discovered the simple approach, which will be stated below, and with his excitement he carved the formula in the stone of a bridge crossing the canal. Today a plaque marks this spot.

Since

$ij = k$, $jk=i$, $ki = j$;

Therefore;

$(ij)k = kk = k^2 = -1$,

$(jk)i = ii = i^2 = -1$,

$(ki)j = jj = j^2 = -1$

as also

$i(jk) = ii = i^2 = -1$,

$j(ki) = jj = j^2 = -1$,

$k(ij) = kk = k^2 = -1$.

We therefore can omit the parenthesis and write

$$ijk = jki = kij = -1 \qquad \text{(Eq.6)}$$

c. Representation of Quaternions by Quadrantal Versors

By using the results in Equation 6 and Table 1, quaternions can be denoted as quadrinomial expressions, of which one term is called the real part, while the three other terms made up together a trinomial, which was called the imaginary part of the quaternion: the square of the former part (or term) being always a positive, but the square of the latter part (or trinomial) being always a negative quantity. More particularly, this imaginary trinomial was of the form $ix + jy + kz$, in which x, y, z were three real and independent coefficients, or constituents, and were, in several applications of the theory, constructed or represented by three rectangular coordinates; while i, j, k were certain imaginary units, or symbols, subject to the following laws of combination as regards their squares and products.[WILKINS99]

d. Identical Quaternions

Assume $\mathbf{q_1}$ and $\mathbf{q_2}$ are two quaternions.

$$\mathbf{q_1} = \mathbf{q_2}, \qquad \text{(Eq.7)}$$

in which

$$q_1 = w_1 + ix_1 + jy_1 + kz_1, \qquad \text{(Eq.8)}$$

and

$$q_2 = w_2 + ix_2 + jy_2 + kz_2 \qquad \text{(Eq.9)}$$

Implies

$w_1 = w_2$ $x_1 = x_2$ $y_1 = y_2$ and $z_1 = z_2$.

e. Addition and Subtraction of Quaternions

Quaternions are added or subtracted by adding or subtracting their constituents, so that

$$\mathbf{q_1} + \mathbf{q_2} = (w_1 + w_2) + i(x_1 + x_2) + j(y_1 + y_2) + k(z_1 + z_2) \qquad \text{(Eq.10)}$$

f. Multiplication of Quaternions

Multiplication for the primitive elements i, j, and k is defined in Equation 6.

Multiplication of quaternions is defined by

$\mathbf{q_0 q_1} = (w0 + x0i + y0j + z0k)(w1 + x1i + y1j + z1k)$

$= (w_0w_1 - x_0x_1 - y_0y_1 - z_0z_1)$

$(w_0x_1 + x_0w_1 + y_0z_1 - z_0y_1)i+$

$(w_0y_1 - x_0z_1 + y_0w_1 + z_0x_1)j+$

$(w_0z_1 + x_0y_1 - y_0x_1 + z_0w_1)k.$ (Eq.11)

Multiplication is not commutative in that the products $\mathbf{q_0q_1}$ and $\mathbf{q_1q_0}$ are not necessarily equal. [MORSE53]

A simple but useful function is the selection function

$W(\mathbf{q}) = W(w + xi + yj + zk) = w$

which selects the “real part” of the quaternion.

The quaternion $\mathbf{q} = w + xi + yj + zk$ may also be viewed as

$$q = w + \vec{v}$$
$$\vec{v} = xi + jy + kz$$

.

If we identify $\vec{v}$ with the 3D vector (x, y, z), then quaternion multiplication can be written using vector dot product (•) and cross product (×) as

$$(w_0 + \vec{v}_0)(w_1 + \vec{v}_1) = (w_0 w_1 - v_0 \bullet v_1) + w_0 \vec{v}_1 + w_1 \vec{v}_0 + \vec{v}_0 \times \vec{v}_1 \qquad \text{(Eq.12)}$$

In this form it is clear that $\mathbf{q_0 q_1} = \mathbf{q_1 q_0}$ if and only if $\vec{v}_0 \times \vec{v}_1 = 0$ (i.e. vectors are parallel). [EBERLY99]

A quaternion q may also be viewed as a 4D vector (w, x, y, z). The dot product of two quaternions is

$$\mathbf{q_0} \bullet \mathbf{q_1} = w_0 w_1 + x_0 x_1 + y_0 y_1 + z_0 z_1 = W(\mathbf{q_0 q_1}^*). \qquad \text{(Eq.13)}$$

g. Conjugate and Norm of a Quaternion

The *conjugate* of a quaternion is defined by

$$\mathbf{q}^* = (w + xi + yj + zk)^* = w - xi - yj - zk. \qquad \text{(Eq.14)}$$

Therefore $(\mathbf{p}^*)^* = \mathbf{p}$

The conjugate of a product of quaternions satisfies the properties

$(\mathbf{pq})^* = q^* p^*$.

The *norm* of a quaternion is defined by

$$N(\mathbf{q}) = N(w + xi + yj + zk) = w^2 + x^2 + y^2 + z^2. \qquad \text{(Eq.15)}$$

Therefore $N(\mathbf{q}^*) = N(\mathbf{q})$

The norm is a real-valued function and the norm of a product of quaternions satisfies the properties

$N(\mathbf{pq}) = N(\mathbf{p})N(\mathbf{q})$.

h. Multiplicative Inverse of a Quaternion

The *multiplicative inverse* of a quaternion **q** is denoted $\mathbf{q}^{-1}$ and has the property $\mathbf{q}\mathbf{q}^{-1} = \mathbf{q}^{-1}\mathbf{q} = 1$. It is constructed as

$$\mathbf{q}^{-1} = \mathbf{q}^{*} / N(\mathbf{q}) \qquad \text{(Eq.16)}$$

Where the division of a quaternion by a real-valued scalar is just component wise division. The inverse operation satisfies the properties

$$(\mathbf{p}^{-1})^{-1} = p \text{ and } (\mathbf{pq})^{-1} = \mathbf{q}^{-1}\mathbf{p}^{-1}. \qquad \text{(Eq.17)}$$

i. Unit Quaternion and Exponential Form of a Quaternion

A unit quaternion is a quaternion **q** for which N(**q**) = 1. The inverse of a unit quaternion and the product of unit quaternions are themselves unit quaternions. A unit quaternion can be represented by

$$\mathbf{q} = \cos\theta + \vec{u}\sin\theta \qquad \text{(Eq.18)}$$

where $\vec{u}$ as a 3D vector has length 1. However, observe that the quaternion product

$\vec{u}\,\vec{u} = -1$. Note the similarity to unit length complex numbers

$\cos\theta + \sin\theta$. In fact, Euler's identity for complex numbers generalizes to quaternions,

$$\exp(\vec{u}\theta) = \cos\theta + \vec{u}\sin\theta \qquad \text{(Eq.19)}$$

where the exponential on the left-hand side is evaluated by symbolically substituting $\hat{u}\,\theta$ into the power series representation for exp(x) and replacing products $\vec{u}\,\vec{u}$ by - 1. From this identity it is possible to define the power of a unit quaternion,

$$\mathbf{q}^{t} = (\cos\theta + \vec{u}\sin\theta)^{t} = \exp(\vec{u}t\theta) = \cos t\theta + \vec{u}\sin t\theta. \qquad \text{(Eq.20)}$$

It is also possible to define the logarithm of a unit quaternion,

$$\log(\mathbf{q}) = \log(\cos\theta + \vec{u}\sin\theta) = \log(\exp(\vec{u}\theta)) = \vec{u}\theta. \qquad \text{(Eq.21)}$$

It is important to note that the noncommutativity of quaternion multiplication disallows the standard identities for exponential and logarithm functions.

The quaternions exp(**p**) exp(**q**) and exp(**p + q**) are not necessarily equal. The quaternions log(**pq**) and log(**p**) + log(**q**) are not necessarily equal.

3. Application of Quaternions to Geometric Rotations

A unit quaternion **q** = cos θ + $\vec{u}$ sin θ represents the rotation of the 3D vector $\vec{v}$ by an angle 2θ about the 3D axis $\vec{u}$. The rotated vector, represented as a quaternion, is R($\vec{v}$) = $\mathbf{q}\vec{v}\mathbf{q}^*$ .The proof requires showing that R($\vec{v}$) is a 3D vector, a length–preserving function of 3D vectors, a linear transformation, and does not have a reflection component.

To see that R($\vec{v}$) is a 3D vector,

$W(R(\vec{v}))= W(\mathbf{q}\vec{v}\mathbf{q}^*)$

$= [(\mathbf{q}\vec{v}\mathbf{q}^*) + (\mathbf{q}\vec{v}\mathbf{q}^*)^*]/2$

$= [\mathbf{q}\vec{v}\mathbf{q}^* + \mathbf{q}\vec{v}^*\mathbf{q}^*]/2$

$= \mathbf{q}[(\vec{v}+\vec{v}^*)/2]\mathbf{q}^*$

$= \mathbf{q}W(\vec{v})\mathbf{q}^*$

$= W(\vec{v})$

$= 0.$

To see that R($\vec{v}$) is length–preserving,

$N(R(\vec{v})) = N(\mathbf{q}\vec{v}\mathbf{q}^*)$

$= N(\mathbf{q})N(\vec{v})N(\mathbf{q}^*)$

$= N(\mathbf{q})N(\vec{v})N(\mathbf{q}^*)$

$= N(\vec{v}).$

To see that R($\vec{v}$) is a linear transformation, let α be a real–valued scalar and let $\vec{v}$ and $\vec{w}$ be 3D vectors; then

$R(\alpha\vec{v} + \vec{w}) = \mathbf{q}(\alpha\vec{v} + \vec{w})\mathbf{q}^*$

$= (\mathbf{q}\alpha\vec{v}\,\mathbf{q}^*) + (\mathbf{q}\;\vec{w}\;\mathbf{q}^*)$

$= \alpha(\mathbf{q}\vec{v}\;\mathbf{q}^*) + (\mathbf{q}\;\vec{w}\;\mathbf{q}^*)$

$= \alpha R(\vec{v}) + R(\vec{w})$,

thereby showing that the transform of a linear combination of vectors is the linear combination of the transforms. The previous three properties show that $R(\vec{v})$ is an orthonormal transformation. Such transformations include rotations and reflections. Consider R as a function of **q** for a fixed vector $\vec{v}$. That is, $R(\mathbf{q}) = \mathbf{q}\vec{v}\,\mathbf{q}^*$.

This function is a continuous function of **q**. For each **q** it is a linear transformation with determinant D(**q**), so the determinant itself is a continuous function of **q**. Thus,

$\lim_{\mathbf{q}\to 0} R(\mathbf{q}) = I$, the identity function

(the limit is taken along any path of quaternions which approach the zero quaternion)

$\lim_{\mathbf{q}\to 0} D(\mathbf{q}) = D(0) = 1.$

By continuity, D(**q**) is identically 1 and R(**q**) does not have a reflection component. Now we prove that the unit rotation axis is the 3D vector $\vec{u}$ and the rotation angle is 2θ. To see that $\vec{u}$ is a unit rotation axis we need only show that $\vec{u}$ is unchanged by the rotation. Recall that $\vec{u}^2 = \vec{u}\,\vec{u} = -1$, which implies that $\vec{u}^3 = -\vec{u}$.

$R(\vec{u}) = \mathbf{q}\vec{u}\,\mathbf{q}^*$

$= (\cos\theta + \vec{u}\sin\theta)\vec{u}\,(\cos\theta - \vec{u}\sin\theta)$

$= (\cos\theta)^2\,\vec{u} - (\sin\theta)^2\,\vec{u}^3$

$= (\cos\theta)^2\,\vec{u} - (\sin\theta)^2\,(-\vec{u})$

$= \vec{u}$.

To see that the rotation angle is 2θ, let $\vec{u}$, $\vec{v}$, and $\vec{w}$ be a right–handed set of orthonormal vectors. That is, the vectors are all of unit length $\vec{u} \bullet \vec{v} = \vec{u} \bullet \vec{w} = \vec{v} \bullet \vec{w} = 0$; and $\vec{u} \times \vec{v} = \vec{w}$, $\vec{v} \times \vec{w} = \vec{u}$, and $\vec{w} \times \vec{u} = \vec{v}$. The vector $\vec{v}$ is rotated by an angle ϕ to the vector $\mathbf{q}\vec{v}\,\mathbf{q}^*$, so $\vec{v} \bullet (\mathbf{q}\vec{v}\,\mathbf{q}^*) = \cos(\phi)$. Using equation (13) and $\vec{v} = -\vec{v}$, and $\vec{p}^{\,2} = -1$ for unit quaternions with zero real part,

$$\cos(\phi) = \vec{v} \bullet (\mathbf{q}\vec{v}\,\mathbf{q}^*)$$

$$= W(\vec{v}^{\,*}\mathbf{q}\vec{v}\,\mathbf{q}^*)$$

$$= W[-\vec{v}\,(\cos\theta + \vec{u}\sin\theta)\vec{v}\,(\cos\theta - \vec{u}\sin\theta)]$$

$$= W[(-\vec{v}\cos\theta - \vec{v}\,\vec{u}\sin\theta)(\vec{v}\cos\theta - \vec{v}\,\vec{u}\sin\theta)]$$

$$= W[-\vec{v}^{\,2}(\cos\theta)^2 + \vec{v}^{\,2}\,\vec{u}\sin\theta\cos\theta - \vec{v}\,\vec{u}\,\vec{v}\sin\theta\cos\theta + (\vec{v}\,\vec{u})^2(\sin\theta)^2]$$

$$= W[(\cos\theta)^2 - (\sin\theta)^2 - (\vec{u} + \vec{v}\,\vec{u}\,\vec{v})\sin\theta\cos\theta]$$

Now $\vec{v}\,\vec{u} = -\vec{v} \bullet \vec{u} + \vec{v} \times \vec{u} = \vec{v} \times \vec{u} = -\vec{w}$ and $\vec{v}\,\vec{u}\,\vec{v} = -\vec{w}\,\vec{v} = \vec{w} \bullet \vec{v} - \vec{w} \times \vec{v} = \vec{u}$. Consequently,

$$\cos(\phi) = W[(\cos\theta)^2 - (\sin\theta)^2 - (\vec{u} + \vec{v}\,\vec{u}\,\vec{v})\sin\theta\cos\theta]$$

$$= W[(\cos\theta)^2 - (\sin\theta)^2 - \vec{u}\,(2\sin\theta\cos\theta)]$$

$$= (\cos\theta)^2 - (\sin\theta)^2 = \cos(2\theta)$$

and the rotation angle is $\phi = 2\theta$.

It is important to note that the quaternions $\mathbf{q}$ and $-\mathbf{q}$ represent the same rotation since $(-\mathbf{q})\,\vec{v}\,(-\mathbf{q})^* = \mathbf{q}\vec{v}\,\mathbf{q}^*$.[SHOEMAKE87]

While either quaternion will do, the interpolation methods require choosing one over the other.

B. SOURCELESS MARG SENSOR BODY TRACKING SYSTEM

1. Current Body Tracking Systems

In general, position and orientation tracking has seen inadequate innovation and development over the past decade. This continues to obstruct advanced development of immersive systems that allow participants to enter and navigate simulated environments. Today’s commercial motion tracking systems are based on optical, magnetic and acoustic sources. Inertial sensing has been used for head tracking. RF positioning shows promise, but no small-scale commercial systems are currently available for indoor use. The most popular trackers are active AC or DC magnetic systems. [BACHMANN2000].

a. Mechanical Trackers

Mechanical tracking systems are perhaps the oldest motion tracking technology. They offer the best means of providing haptic feedback to the user of **a** virtual environment. These systems are fairly accurate and have low latency. Current research generally involves using these tracking systems to calibrate other types of trackers.[BACHMANN2000]

Most famous example of mechanical body tracking system is Exoskeleton.

b. Optical Trackers

Optical trackers are based on tracking of luminescent object by special optic sensors. (Fig.3) This technology is known as Instant Marker Recognition Technology (IMR). These systems usually consist of a cage, IR-sensors, and data processing computer. There are hundreds of digital sensors and electronics, which are embedded in the bars of cage. They employ IMR technology to cleanly track markers. Markers send optical signals for different time intervals. Since the cameras are located in fixed modular bars, a blocked marker is instantly reacquired once occlusion sends. This produces consistently clean data streams with minimal data loss. (Fig.3)

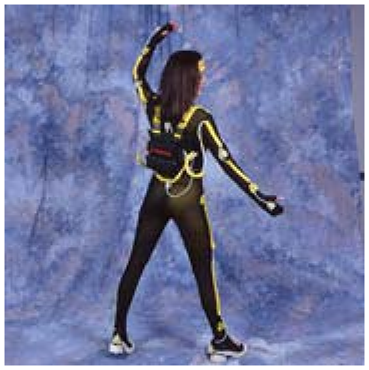


Figure 3. Optical Markers (Courtesy of Ascension-tech inc 2001)

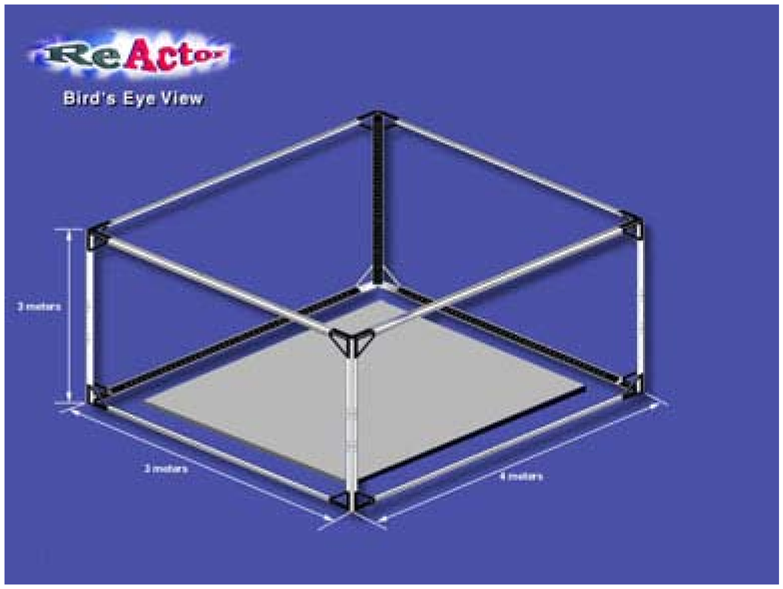
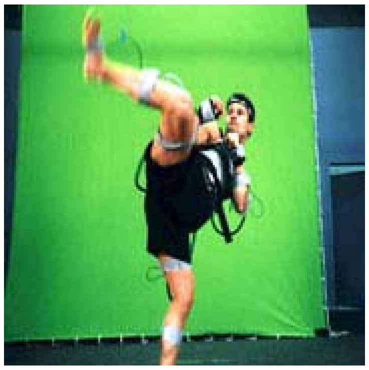


Figure 4. Optical Motion Capture Cage. Courtesy of Ascension-tech Inc. 2001

The disadvantages of optical motion capture systems are occlusion of body parts and limited motion capture area (locality).

c. *Magnetic Trackers*

Magnetic trackers measure the position and orientation of one or more receiving antenna sensors, typically located on a user's head, hand or body, with respect to a transmitting antenna, which is fixed in space. The transmitting antenna is driven by a pulsed, direct current (DC) signal.



The receiving antenna measures not only the transmitted magnetic field pulse but also the earth's magnetic field. A microprocessor is used to control the transmitting and receiving elements and convert the received signals into position and orientation outputs. (Fig.5)

Figure 5. Motion Capture with Magnetic Trackers

Electromagnetic trackers suffer from several sources of error. Electromagnetic interference (EMI) from devices such as radios or display units can cause erroneous readings. Large objects made of ferrous metals can interfere with the electromagnetic field, again causing inaccuracies [HAND93].

Robustness is adversely effected by sensitivity to ferromagnetic objects in the vicinity, with alternating current based trackers being more susceptible than direct current based trackers. Alternating current systems tend to generate eddy currents in metallic objects, which then cause their own electromagnetic interference [SKOP96].

d. Fiberoptic Cable Deformation Based Motion Capture

This system is based on light refraction measurement techniques. When the light travels through Fiberoptic cable it has a constant energy (frequency). But deformation of cable causes reflection and refraction of this beam inside the cable. And one can measure this energy changes and correlate with the deformation angle.

ShapeTape © is patented technology which uses this principle.(Fig.6)

ShapeTape is a light weight, wearable, flexible ribbon that uses its own software, ShapeWare, to create a 3D computer image and data set of its shape in real time, based on bend and twist information from an array of fiber optic sensors along its length. It follows human arm, leg, back, and neck movements for motion capture, virtual reality, biomedical, gaming, and robotic control applications. It is also used for crash testing, computer-aided design, and automotive interior design

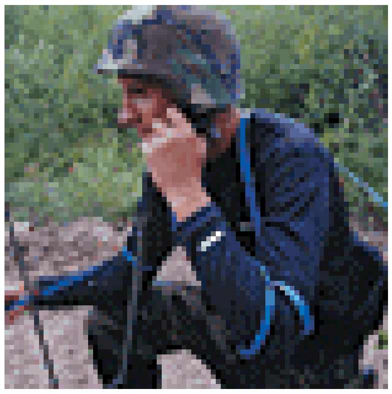


Figure 6. Shape Tape©

.

e. Acoustic Trackers

Acoustic or ultrasonic trackers are an inexpensive option to magnetic trackers. They offer modest accuracies and update rates.

The physics of sound limit the accuracy, update rate and range of acoustic tracking systems. Ranges are longer than that of magnetic trackers and magnetic interference is not a problem. However, a clear line of sight must be maintained. Thus, obstruction and shadowing can present difficulties. Latency varies with distance due to the relatively slow speed of sound. Most current systems utilize 40 kHz tone pulses. Sound in this frequency band can be rigorously affected by noise from metallic objects such as key chain. Shorter wavelengths more precisely resolve distances, but quickly attenuate. In addition, high frequency omnidirectional radiators are expensive to implement and require more power.

f. Pattern Recognition

Passive stereo vision systems employ ambient light and square-array charge-coupled device (CCD) cameras [DURL95]. Multiple images from cameras with varying viewpoints are compared. Triangulation is then used to determine position [SKOP96]. An essential issue is to solve the correspondence problem of relating the same points in two different images. Passive stereo vision systems are unlikely to be useful in virtual environment in the near term, as robustness and accuracy are not yet comparable to active ranging systems [DURL95].

2. Fundamental Principles of MARG Body Tracking System

For real-time human limb segment tracking, most current methods use active sources of some sort to track either angles or the position of reference points on limbs. While this has worked well for some applications such approaches typically either seriously impede the objects or work only for local areas.[YUN99].

Recent advances in micro machine and embedded electronic technology have made it possible to design a sourceless sensor system. This system uses earth magnetic and gravitational field as references in order to determine the orientation of an object. An example of this system is explained in detail for unmanned submarine navigation in [YUN99].

MARG Body Tracking system uses the same idea with extended sensor collection. They are magnetic and gravitational field of earth and angular rates of body segments.

a. Quaternion Filter

MARG Body tracking system uses quaternion angle sets to determine the orientation. Because quaternions are free from singularity problems of Euler angles and much more cheaper for rate transformations.[MCGHEE99]

Let p, q, r be roll rate, pitch rate and yaw rate respectively [COOKE92], and let **q** = (0 p**i** q**j** r**k**) Then

$$\dot{Q} = \frac{1}{2} Q \otimes (q). \qquad \text{(Eq.22)}$$

We can have an exact orientation by integrating equation 22. But such an accurate and small rate sensor is not available for body tracking in today's technology. Instead, by using micro machine technology, vibrating beam sensors are more suitable for this purpose. Because, they are small, cheap, and require negligible electric power. But their error rates become unacceptable after only few seconds of integration. For preventing this [BACHMANN99] shows a system with drift correction. Figure 7 is the quaternion filter, which is used in our MARG body tracking system.

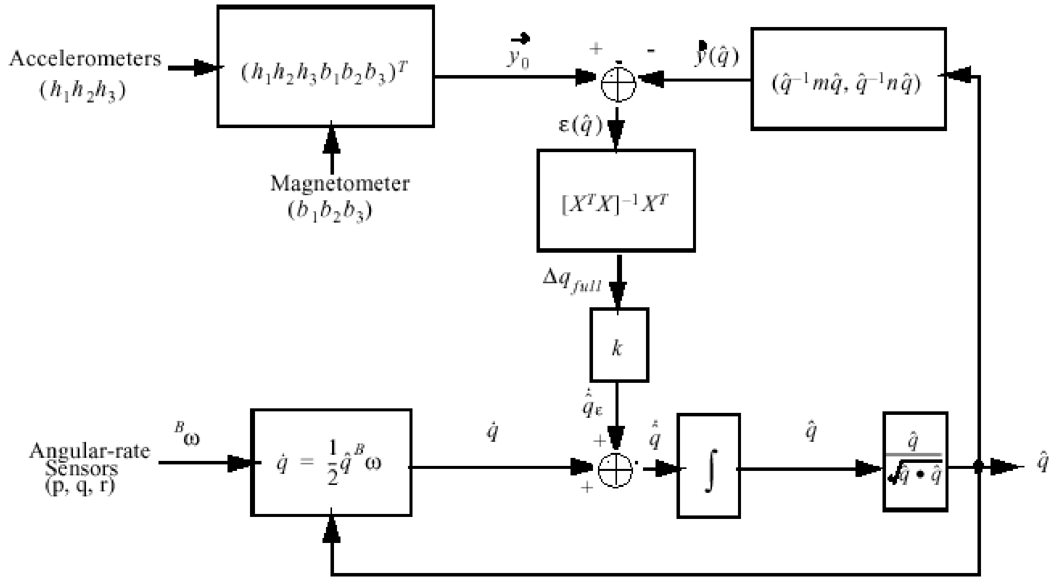


Figure 7. Quaternion based orientation filter.

Drift correction is provided by Accelerometers and Magnetometers. [YUN2000] explained how to find modeling error and estimate quaternion orientation by non-linear regression analysis in detail.

C. VIRTUAL HUMANS IN VIRTUAL ENVIRONMENTS

1. Purpose of a Virtual Human in Cyberspace

Old religious scripts say that whole universe is created for humans. New scriptures say that whole Cyberspace is created for Virtual Humans. The main purpose of a virtual human is to represent human existence within the virtual world.

The main reason for developing realistic Virtual Human is to be able to use them in virtually any scene [THALMANN97]. Potential human activities, which may involve humanoids, are as follows

- Simulation based learning and training
- Simulation of ergonomic work environments
- Virtual patient for surgery
- Orthopedy and prostheses and rehabilitation
- Architectural simulation with people
- Computer games involving people
- Game and sport simulation
- Interactive drama titles (play with interacting virtual characters)
- Military training simulations
- Creating virtual terrorist agents with unpredictable reaction in order to prepare safety measures.
- Crowd simulations
- Virtual meetings

Virtual Humans will represent the people in different virtual environments. Telepresence is the future of multimedia systems. This will allow people to share experiences and interact with each other via virtual humans. These types of virtual environments may contain human controlled virtual humans or autonomous humanoids or partially intelligent virtual characters.

There are different aspects of virtual human research. Following table shows some of the ongoing projects within the computer graphics community.

Face and body representation
Motion control
Interaction with objects
Lip synchronization
Autonomous behavior
Deformable body shape
Interaction between virtual humans
Networked virtual Environments
Crowds
Rendering
Standard data protocols for body orientations
Interaction by users
Optimized rendering

Table 2. Aspects of Virtual Human Research [From THALMANN97]

Our research deals with construction of virtual human with deformable skin for a motion capture system.

2. Current Techniques For Inserting Virtual Humans Into Virtual Environments

Current techniques for inserting humanoids into virtual worlds are not real-time. They usually have pre-production and postproduction phases. Some of these techniques involve user interaction to avatars by some kind of device such as mouse or joystick. The main problem of these systems is that they don’t represent a realistic human motion.

People are skilled at perceiving the subtle details of human motion. A person is often able to recognize other people at a distance by their walk. Because of this ability, people have high standards for animations that feature humans. For realistic computer-generated motion, the virtual actors must move with a natural-looking style.

Today computer generated human motion is needed for such applications as animation, virtual environments and video games. For example, trainers could use virtual sportsmen to motivate and teach athletes. Video game designers could create products with highly interactive, engaging characters. In the Navy, the movements of a fighting soldier in simulated or real-time environment can be tracked. The ability to simulate human motion also has significant scientific applications in ergonomics, and physical rehabilitation.[HODGINGS2000]

Although the applications for synthetic human motion are many, specifying movement to a computer is surprisingly hard. Even a simple bouncing ball can be difficult to animate convincingly, in part because people quickly pick out action that is unnatural or implausible without necessarily knowing exactly what is wrong. Animation of a human is especially time-consuming because subtleties of the motion must be captured to convey personality and mood.

There are there types of human motion generation:

- Keyframing
- Motion Capture
- Simulation

Keyframing forces the animator to smooth between frames. Simulations generate motion in a fairly automatic fashion but strictly depend on minimization of error between the physical based model of motion and real motion by computers. However body tracking is a discrete sampling of real-time motion and gets as realistic as sampling frequency increases[HODGINS2000].

THIS PAGE INTENTIONALLY LEFT BLANK

III. MODELING A VIRTUAL HUMAN

A. KINEMATIC MODELS

1. Introduction

The virtual human body is typically modeled using a hierarchical structure. Joints are the nodes of this structure and segments such as arms, legs are links between these joints. In order to create a realistic human representation one needs to define kinematic properties of this structure.

Kinematic models specify geometric and time related properties of the links between joints. Position, orientation, velocity and acceleration are examples of properties of motion of each link. Rigid body transformation is a fundamental approach to calculate orientation of each segment. *End-effector* is the final segment of hierarchical structure.

There are two types of Kinematics. *Forward Kinematics* determines the orientation of end-effector by using the properties of each segment. *Inverse kinematics* determines the properties of each segment in order to reach final transformation of end-effector.

This research used H-Anim specification to define kinematic properties of a virtual human. H-Anim is a structural model with segment and joint containers.

2. Rigid Body Orientation

Human posture can be modeled as group of orientations of body segments. Each segment is a rigid body in terms of physics. There are different ways to represent rigid body rotations. This part of this chapter will explain two of them: Euler and quaternion rotations.

a. Euler Angle Rotations

One of the popular methods for representing orientation in Earth coordinates is the Euler Method. Euler method uses simple sequence of three angles. This makes Euler Method an intuitive description. Although eleven other possibilities exist, these angles typically consist of the familiar azimuth angle, ψ, the elevation angle, θ , and the roll angle, φ , [CRAI89]. Euler angles specify successive rotations to bring the Earth coordinates into alignment with the body coordinates [COOK92].

Euler's theorem states that any numbers of rotations of a rigid body about an Earth-fixed axis are equivalent to a single rotation about a single Earth- fixed axis. If all rotations are about the north(x), east(y) and down(z) axes, as depicted in Figure 8 , the angles are called Euler Angles.

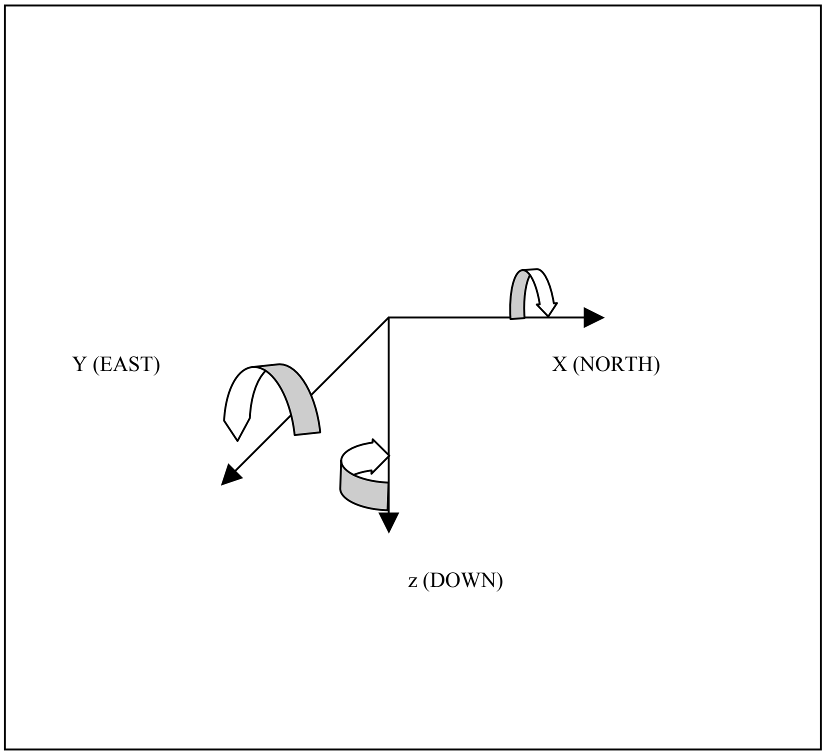


Figure 8. Right handed Euler Rotations (Earth Axis)

The standard "azimuth, elevation, and roll" set is represented by the three rotation matrices given in Equation 23. R_x represents a rotation about the x axis [CRAI89], and similarly for R_y and R_z.

$$R = R_z R_y R_x \qquad \text{(Eq.23)}$$

These symbols are reserved for Euler Rotations with the angles ψ, θ, ϕ.

Right Hand rule determines the direction (sign) of the rotation. Limits of the rotations are as follows:

$$\psi = \pm\pi, \qquad \theta = \pm\frac{\pi}{2}, \qquad \phi = \pm\pi. \qquad \text{(Eq.24)}$$

X-axis rotation (roll) matrix is given by

$$R_x(\varphi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & \sin\varphi \\ 0 & -\sin\varphi & \cos\varphi \end{bmatrix} \qquad \text{(Eq.25)}$$

Y-axis rotation (elevation, pitch) ;

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \qquad \text{(Eq.26)}$$

Z-axis rotation (azimuth, heading, yaw) is given by

$$R_z(\psi) = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \text{(Eq.27)}$$

Every rotation matrix is a 3x3 matrix, but to make both translation and rotation calculations using matrices, 4x4 homogeneous matrices are used. A homogeneous rotation matrix in graphics takes the form [FOLE97]

$$R = \begin{bmatrix} r_1 & r_2 & r_3 & a \\ r_4 & r_5 & r_6 & b \\ r_7 & r_8 & r_9 & c \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{(Eq.28)}$$

Rotation of a vertex is calculated by multiplying these rotation matrices with homogeneous coordinates of this vertex. First rotation is associated with the rightmost matrix. And successive multiplications determine the final orientation of the object. For example a roll followed by an elevation and another azimuth rotation will form matrix R by multiplying corresponding matrices from right to left.

$$R = R_z R_y R_x \qquad \text{(Eq.29)}$$

$$R = \begin{bmatrix} \cos\psi\cos\theta & \sin\varphi\sin\theta\cos\psi - \cos\varphi\sin\psi & \sin\varphi\sin\psi + \cos\varphi\cos\psi\sin\theta \\ \sin\psi\cos\theta & \sin\varphi\sin\theta\sin\psi + \cos\varphi\cos\psi & \cos\varphi\sin\theta\sin\psi - \cos\psi\sin\varphi \\ -\sin\theta & \cos\theta\sin\varphi & \cos\varphi\cos\theta \end{bmatrix} \qquad \text{(Eq.30)}$$

b. Body Rates to Euler Rates

Representing a rotational velocity around the nose vector, side vector(s), and the approach or belly vector requires a standard frame.(Fig.9) Two common coordinate frames are Earth and Body coordinate frames.

Rotational velocity in Earth coordinates is;

$$E_\omega = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^T \qquad \text{(Eq.31)}$$

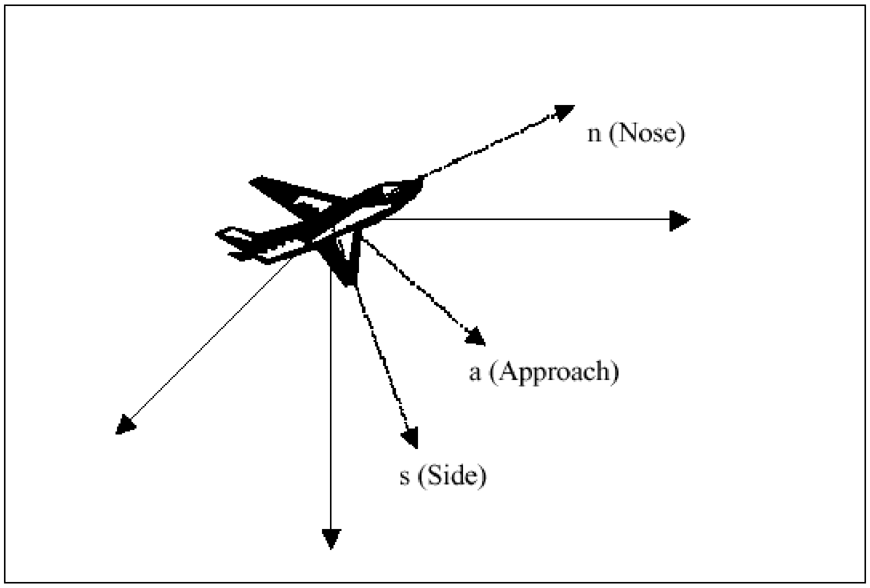


Figure 9. Body Coordinates Axis

And in body coordinates

$$B_\omega = [p \quad q \quad r]^T \tag{Eq.32}$$

where p is roll rate, q is pitch rate and r is yaw rate. These symbols are reserved for body rotations [MCGHEE93]. Euler rates are in Earth coordinates while p.q.r are in body coordinates.

After doing calculation to solve Body coordinates in terms of Earth coordinates [DUMAN2000] we have the following results.

$$\begin{bmatrix} \dot{\varphi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \tan\theta\sin\varphi & \tan\theta\cos\varphi \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sec\theta\sin\varphi & \sec\theta\cos\varphi \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{Eq.33}$$

Equation 33 is known as the gimbal rate equations and the matrix is named as T matrix. Currently this is the most used form for animation and simulations.[FRAN69]

c. Rigid Body Dynamics with Quaternions

Euler angles represent the rotation of rigid body around three orthogonal coordinates. However we can describe the orientation of these objects by a single rotation (θ) around a single vector($\vec{u}$). Moreover we can constraint this vector with a unit magnitude. Thus unit quaternion representing this rotation is [MCGHEE97]

$$q = \begin{pmatrix} \cos\frac{\theta}{2} & \vec{u}\sin\frac{\theta}{2} \end{pmatrix} \qquad \text{(Eq.34)}$$

[SHOEMAKER87] defines the rotation of vector p by quaternion q as follows,

$$\vec{p}_{rotated} = qpq^{-1} \qquad \text{(Eq.35)}$$

The reason of half angle in Equation 34 is that quaternion q in Equation 35 rotates the vectors perpendicular component twice the angle which is perpendicular to rotation axis.

After redefining quaternion rotations now we need to derive quaternions from body rates. Angular rates p ,q, r can be used to form a $\dot{q}$ quaternion rate relative to Earth coordinates.

Assumptions:

$$\cos\frac{\theta}{2} \cong 1 \quad , \quad \sin\frac{\theta}{2} \cong \frac{\theta}{2} \qquad \text{(Eq.36)}$$

Thus

$$q = \begin{pmatrix} \cos\frac{\theta}{2} & \vec{u}\sin\frac{\theta}{2} \end{pmatrix} \cong q = \begin{pmatrix} 1 & \vec{v}\frac{\theta}{2} \end{pmatrix} \qquad \text{(Eq.37)}$$

For a small time change, the rotation quaternion can be expressed as a function of time as follows.

$$q(t) = \begin{pmatrix} 1 & \frac{1}{2}\vec{v}\dot{\theta}t \end{pmatrix} \qquad \text{(Eq.38)}$$

$\vec{v}\dot{\theta}$ is angular rate about a vector $\vec{v}$ in body vectors which is equivalent to body rates p, q, r.

$$\vec{v}\dot{\theta} = (\text{p q r}) \qquad \text{(Eq.39)}$$

and

$$q(t) = \begin{pmatrix} 1 & \frac{1}{2}pt\frac{1}{2}qt\frac{1}{2}rt \end{pmatrix} \qquad \text{(Eq.40)}$$

$$\dot{q}(t) = \begin{pmatrix} 0 & \frac{1}{2}p\frac{1}{2}q\frac{1}{2}r \end{pmatrix}$$

$$\dot{q}(t) = \frac{1}{2}\begin{pmatrix} 0 & p & q & r \end{pmatrix}$$

$$\dot{q}(t) = \frac{1}{2}B_{\omega} \qquad \text{(Eq.41)}$$

Let q_1 be initial orientation in Earth coordinates and q_2 be a second rotation in body coordinates and q_3 be the composite of these two rotations.[BACHMANN2000]

$$q_3 = q_1 q_2$$

Let's derive q$_3$ with respect to time.

$$\dot{q}_3 = \dot{q}_1 q_2 + q_1 \dot{q}_2$$

Derivation of q$_1$ will be zero because it is constant. And by using Equation 41 derivation of q$_2$ can be substituted by its rate values.

$$\dot{q}_3(t) = q_1 \frac{1}{2} B_\omega \qquad \text{(Eq.42)}$$

d. Rotation Matrices From Quaternions

Currently there isn't a computer graphics hardware, which is directly using quaternions for displaying vertices on the screen. This forced virtual human interface to own a converter from quaternion rotation to axis angle rotations. Conversion can be achieved by using following matrix.

Let q = (w + xi + yj + zk) then ,

$$Q_{matrix} = \begin{pmatrix} w^2+x^2-y^2-z^2 & 2xy+2wz & 2xz-2wy & 0 \\ 2xy-2wz & w^2-x^2+y^2-z^2 & 2xy+2wx & 0 \\ 2xz+2wy & 2yz-2wx & w^2-x^2-y^2+z^2 & 0 \\ 0 & 0 & 0 & w^2+x^2+y^2+z^2 \end{pmatrix}$$

Rotated vector can be calculated by multiplication of Q matrix to vector **P**.

3. Comparison Of Quaternion And Euler Angles For Kinematic Models

Quaternions and Euler angles each have their own advantages and disadvantages. The most major advantage of quaternions is that no singularity exists when the elevation angle (θ) passes through $\pm\pi$ /2. In the Euler Method, roll and azimuth Euler angle rates become very large (infinite) due to division by zero. Truncating the angles at $\pm\pi$/2 will avoid this problem. However, this truncating will result in jump during the rotation [COOK92].

Quaternion rotation avoids the “branch cut” problem of Euler angles. When using quaternions to rotate an object, after every full rotation the quaternion representing the rotation will switch to its negative.

Quaternions can be computed directly from the dynamic equations, bypassing the computation of transcendental functions necessary in computing Euler angles. This direct arithmetic operation reduces the cost of computation compared to matrix multiplication.

Quaternions are compact and simple. However, there are three difficulties with using quaternions.

- Each orientation of an object can actually be represented by two quaternions. Since rotation about the axis, v by an angle θ is the same as rotation about $-v$ by the angle $-\theta$; the corresponding quaternions are antipodal points on the sphere in 4D.
- Orientations and rotations are not exactly the same thing. In an animation, 360º rotation is very different from a rotation of 0º. With the first one, the model will be animated for a full rotation. In the second one, no animation will be executed. After the rotation however, the same quaternion (1 0 0 0) or (-1 0 0 0) represents both. Thus specifying multiple rotations with quaternions requires intermediate control points [FOLE97].
- Quaternions provide an isotropic method for rotation. The interpolation is independent of everything except the relation between the initial and final rotations. This is useful for interpolating orientations of tumbling bodies, but not for interpolating the orientations of a virtual camera in a scene. Humans strongly prefer cameras to be upright, and are profoundly disturbed by tilted cameras [FOLE97]. By its very nature, the notion of a preferred direction cannot easily be built into the quaternion representation [WATT98]. Quaternions have no such preferences, and therefore should not be used for camera interpolation. The lack of an adequate method for interpolating complex camera motion has led to many computer animations having static cameras or very limited camera motion [FOLE97].

B. H-ANIM 2.0 STANDARD FOR HUMAN MODELS

1. Introduction

As the 3D Internet continues to grow, there will be an increasing need to characterize human beings in online virtual environments. Achieving that goal will require the establishment of libraries of interchangeable humanoids, as well as authoring tools that make it easy to create new humanoids and animate them in different ways.

H-Anim group specified design goals as follows:

Compatibility: Humanoids should work in any VRML97 compliant browser.

Flexibility: No assumptions are made about the types of applications that will use humanoids.

Simplicity: The specification can always be extended later.

Much of the design is driven by these three goals. The compatibility requirement avoids the use of scripting, since the VRML97 specification does not require any particular scripting language to be implemented. The flexibility requirement makes the spec fairly "low-level", in that it allows direct access to the joints of the humanoid's body and the vertices of the individual body segments. The simplicity requirement focuses specifically on humanoids, instead of trying to deal with arbitrary articulated figures.

The human body consists of a number of segments (such as the forearm, hand and foot), which are connected to each other by joints (such as the elbow, wrist and ankle). In order for an application to animate a humanoid, it needs to obtain access to the joints and alter the joint angles. The application may also need to retrieve information about such things as joint limits and segment masses.

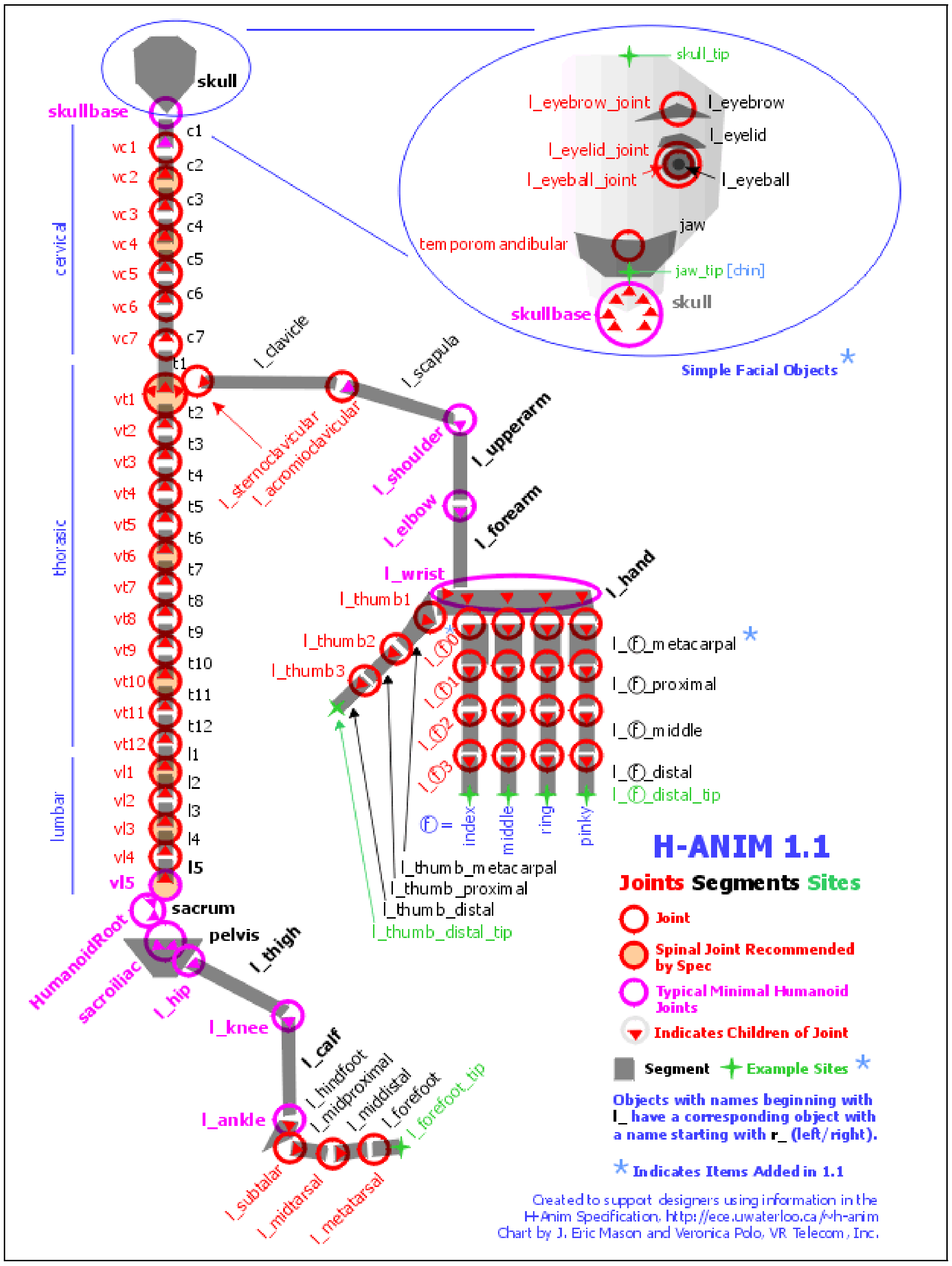


Figure 10. HANIM Joint Hierarchy

A mesh of polygons will typically define each segment of the body, and an application may need to alter the locations of the vertices in that mesh. The application may also need to obtain information about which vertices should be treated as a group for the purpose of deformation.

An H-Anim file contains a set of Joint nodes that are arranged to form a hierarchy. Each Joint node can contain other Joint nodes, and may also contain a Segment node, which describes the body part associated with that joint. Each Segment can also have a number of Site nodes, which define locations relative to the segment. Sites can be used for attaching clothing and jewelry, and can be used as end-effectors for inverse kinematics applications. They can also be used to define eye-points and viewpoint locations.

Each Segment node can have a number of Displacer nodes that specify which vertices within the segment correspond to a particular feature or configuration of vertices. [HANIM2001]

2. Fundamental Prototypes For H-Anim 2.0 Standard

There are five fundamental nodes in H-anim 2001 spec. Here I explained only 3 of them. Because my implementation used only Humanoid , joint , and segment nodes.

a. Humanoid Node

The Humanoid node serves as the overall container for the Joint, Segment, Site and Viewpoint nodes, which define the skeleton, geometry and landmarks of the humanoid figure. Additionally, the node provides a means for defining information about the author, copyright and usage restrictions of the model. Later in this document more specific details about the interface for this node will be described, but for now we will give a general overview of the entities involved and the different ways in which the humanoid figure can be defined.

The geometry of a H-Anim humanoid figure can be described in two ways. The first way is within the scene graph of the joint hierarchy, which is described in the skeleton field of the Humanoid node. Geometry (in the form of Shape nodes) defined within Segment nodes of this joint hierarchy describes the body as separate geometric pieces. This method, while computationally efficient, has certain visual anomalies (such as seams or creases), which detract from the appearance of the humanoid figure. The second way of describing the geometry of an H-Anim humanoid figure is as a continuous piece of geometry, within the skin field of the Humanoid node. For this method, point and normal vector data sets are first defined in the **skinCoord** and **skinNormal** fields (in the form of Coordinate and Normal nodes, respectively). This data is defined in this manner as to allow it to be referenced by two different entities within the definition of the Humanoid node. The first entity is an **IndexedFaceSet** node(s), which defines the surface of the geometry of the humanoid figure within the skin field. In most cases this surface will be a single IndexedFaceSet node, however the surface can also be defined as multiple IndexedFaceSet nodes. It is possible that depending on the implementation of the IndexedFaceSet nodes and the configuration of the humanoid figure, multiple IndexedFaceSet nodes may provide better performance by isolating the continuous mesh changes to localized surfaces.

A skeletal hierarchy is defined for each H-Anim humanoid figure within the skeleton field of the Humanoid node. This hierarchical definition of joints is present even when the geometry of the humanoid figure is not defined within the skeleton field, as the Joint definitions are still needed to describe the coordinate frame about which the vertices of the continuous mesh are deformed. The H-Anim specification's only requirement is that this skeletal hierarchy contains the **HumanoidRoot** Joint. All of the other Joint nodes are optional and are not required for a humanoid figure to be H-Anim compliant. Naturally, most H-Anim figures will have many more joints defined than this required one.

```
PROTO Humanoid [
  field          SFVec3f    bboxCenter          0 0 0
  field          SFVec3f    bboxSize           -1 -1 -1
  exposedField   SFVec3f    center              0 0 0
  exposedField   MFString   info               [ ]
  exposedField   MFNode     joints             [ ]
  exposedField   SFString   name               ""
  exposedField   SFRotation rotation            0 0 1 0
  exposedField   SFVec3f    scale              1 1 1
  exposedField   SFRotation scaleOrientation    0 0 1 0
  exposedField   MFNode     segments           [ ]
  exposedField   MFNode     sites              [ ]
  exposedField   MFNode     skeleton           [ ]
  exposedField   MFNode     skin               [ ]
  exposedField   SFNode     skinCoord           NULL
  exposedField   SFNode     skinNormal          NULL
  exposedField   SFVec3f    translation         0 0 0
  exposedField   SFString   version            "2.0"
  exposedField   MFNode     viewpoints         [ ]
]
```

Figure 11. Humanoid Node Prototype

The bboxCenter and bboxSize fields specify a bounding box that encloses the children of the Humanoid node. This is a hint that may be used for optimization purposes. A default bboxSize value, (-1, -1, -1), implies that the bounding box is not specified and, if needed, shall be calculated by the browser.

The center field specifies a translation offset from the origin of the local coordinate system (0,0,0). The rotation field specifies a rotation of the coordinate system of the humanoid figure. The scale field specifies a non-uniform scale of the humanoid figure coordinate system and the scale values must be greater than zero. The scaleOrientation specifies a rotation of the coordinate system of the humanoid figure before the scale (to specify scales in arbitrary orientations). The scaleOrientation applies only to the scale operation. The translation field specifies a translation to the coordinate system of the entire humanoid figure.

The name field, which must be present, stores the name of the humanoid defined by the Humanoid node.

The skeleton field contains the HumanoidRoot Joint node. The Humanoid node is considered the parent node of the HumanoidRoot Joint node and defines a coordinate space for the HumanoidRoot Joint node. This means that the transformation of the Humanoid node accumulates down the scene graph defined within the skeleton field. As was mentioned in the introduction of this section hierarchy of Joint nodes are defined for each H-Anim humanoid figure within the skeleton field of the Humanoid node and hierarchical definition of joints is present even when the geometry of the humanoid figure is not defined within the skeleton field.

The joints field contains a list of references, one for each Joint node defined within the skeleton field hierarchy of the Humanoid node. This field was added to the H-Anim specification because of a functional limitation with VRML97 when querying the system for a list of all the Joint nodes defined. It is possible that in future versions of the H-Anim specification this list will not be necessary, however for now all implementations of H-Anim humanoid figures are required to implement this feature. The order in which the joints are listed is irrelevant, since the names of the joints are stored in the Joint nodes themselves.

The segments field contains a list of references, one for each Segment node defined within the skeleton field hierarchy of the Humanoid node. This field was added to the H-Anim specification because of a functional limitation with VRML97 when querying the system for a list of all the Segment nodes defined. It is possible that in future versions of the H-Anim specification this list will not be necessary, however for now all implementations of H-Anim humanoid figures are required to implement this feature. The order in which the segments are listed is irrelevant, since the names of the segments are stored in the Segment nodes themselves.

The sites field contains a list of references, one for each Site node defined within the skeleton field hierarchy of the Humanoid node. This field was added to the H-Anim specification because of a functional limitation with VRML97 when querying the system for a list of all the Site nodes defined. It is possible that in future versions of the H-Anim specification this list will not be necessary, however for now all implementations of H-Anim humanoid figures are required to implement this feature. The order in which the sites are listed is irrelevant, since the names of the sites are stored in the Site nodes themselves.

The viewpoints field has a different functionality and behavior than the joints, segments and sites fields. The viewpoints field can contain zero or more Viewpoint node definitions. The Viewpoint nodes defined within this field are affected by the transformations applied to the Humanoid node, but are not affected by any of the transformations performed on the skeletal hierarchy defined within the skeleton field. The order in which the Viewpoint nodes are listed will determine what order they appear in the Viewpoint stack.

The skin field contains a single or a collection of IndexedFaceSet nodes. Those IndexedFaceSet nodes reference the Coordinate and Normal nodes defined within the skinCoord and skinNormal fields, respectively, of the Humanoid node. The Coordinate and Normal nodes are defined in this manner so that the data within them can be accessed by two entities. The first entity is the Humanoid node itself, which has internal mechanisms that can manipulate the Coordinate and Normal data sets to create the appropriate surface deformations as the Joint nodes of the humanoid figure are animated. The second entity that uses these data sets is the IndexedFaceSet nodes defined within the skin field, which describes the surface geometry of the humanoid figure.

The skinCoord field contains a single Coordinate node definition, which is used by the internal mechanisms of the Humanoid node to create the appropriate surface deformations as well as the IndexedFaceSet node which does the actual rendering of the surface geometry.

The skinNormal field contains a single Normal node definition which is used by the internal mechanisms of the Humanoid node to create the appropriate surface deformations as well as the IndexedFaceSet node, which contains the actual surface geometry that is rendered.

The Humanoid node should be given a DEF name of "Humanoid", in order to make it easy to access using the External Authoring Interface.

b. Segment Node

The Segment node is used describe the attributes of the physical links between the joints of the humanoid figure. Each body part (pelvis, thigh, calf, etc) of the humanoid figure is represented by a Segment node. These nodes are organized in their skeletal hierarchy of the humanoid and provide a container for information which is specific to each segment of the body.

The Segment node is a specialized grouping node that provides a container for nodes in it's children field. See the VRML97 specification, specifically the 4.6.5, Grouping and children nodes section, for more details on grouping nodes. The reason a Segment node is considered a specialized grouping node is that it can only be defined as a child of an H-Anim Joint node. To do otherwise would be a violation of the structure of the H-Anim Specification.

It is suggested, but not required, that all of the body segments should be built in place. That is, they should require no translation, rotation, or scaling to be connected with their neighbors. In such a scenario all the body's coordinates will share a common origin, which is that of the humanoid itself. If this proves difficult for an authoring tool to implement, it is acceptable to use a Transform node inside each Segment, or even several Transform nodes, in order to position the geometry for that segment correctly. The reason it is best to avoid such transformations is that it will impact runtime performance. It is also recommended that LOD nodes are used to describe the geometry of Segment nodes of high complexity.

This section describes the interface of the Segment node and the fields which allow the functionality of the node to be defined and accessed.

```
PROTO Segment [
  field          SFVec3f    bboxCenter       0 0 0
  field          SFVec3f    bboxSize         -1 -1 -1
  exposedField   SFVec3f    centerOfMass     0 0 0
  exposedField   MFNode     children         [ ]
  exposedField   SFNode     coord            NULL
  exposedField   MFNode     displacers       [ ]
  exposedField   SFFloat    mass             0
  exposedField   MFFloat    momentsOfInertia [ 0 0 0 0 0 0 0 0 0 ]
  exposedField   SFString   name             ""
  eventIn        MFNode     addChildren
  eventIn        MFNode     removeChildren
]
```

Figure 12.   Segment Node Prototype

The bboxCenter and bboxSize fields specify a bounding box that encloses the children of the Segment node. This is a hint that may be used for optimization purposes. A default bboxSize value, (-1, -1, -1), implies that the bounding box is not specified and, if needed, shall be calculated by the browser.

The name field must be present, so that the Segment can be identified at runtime. The name field is used for identifying the Segment nodes at runtime. Each Segment should have a DEF name that matches the name field for that Segment, but with a distinguishing prefix in front of it. That prefix can be anything, but must be the same for all the Segment nodes in a particular humanoid. The distinguishing prefix is useful in the case of static routing to the Segment nodes of multiple humanoids in the same file. If only a single humanoid is stored in a file, the prefix should be "hanim_" (for Humanoid Animation). For example, the left thigh would have a DEF name of "hanim_l_thigh". The DEF name is used for static routing and the name field, which must be present, is used by applications to identify the Segment node at runtime. All the other fields are optional, but must be defined in the PROTOtype.

The mass field is the total mass of the segment.

The centerOfMass field is the location within the segment of its center of mass. Note that a zero value was chosen for the mass in order to give a clear indication that no mass value is available.

The momentsOfInertia field contains the moment of inertia matrix. The first three elements are the first row of the 3x3 matrix, the next three elements are the second row, and the final three elements are the third row.

The coord field is used for Segment nodes that have deformable meshes and should contain a Coordinate node that is USEd in the IndexedFaceSet for the Segment. The Coordinate node should be given the same name DEF name as the Segment, but with a "_coords" appended (e.g. "skull_coords").

The displacers field stores the Displacer nodes for a particular Segment node.

c. Joint Node

The Joint node is used as a building block to describe the articulations of the humanoid figure. A Joint node, each of which is organized into a hierarchy that describes the overall skeleton of the humanoid, represents each articulation of the humanoid figure. This hierarchy describes the inherent parent-child relationship of the skeleton as well as providing a container for information, which is specific to each joint of the skeleton. Within this description, we will give more specific details about the interface for this node, but for now we will give a general overview of the entities involved and the different ways in which the Joint nodes interact with the geometry of the overall humanoid figure. Before attempting to understand the operation of the Joint node, the reader should become familiar with the Humanoid node and its operation..

The Joint node is a specialized grouping node that defines a coordinate system for nodes in its children field that is relative to the coordinate systems of its parent node. The reason a Joint node is considered a specialized grouping node is that it can only be defined as a child of another Joint node or in the case of the HumanoidRoot Joint, as the first node of the skeletal hierarchy defined in the skeleton field of the Humanoid node. Additionally, a Joint node has two fields, which allow it to manipulate individual vertices from the skinCoord field of the Humanoid node. Incoming rotation field events of the Joint node affect the vertices indicated by the skinCoordIndex field by a factor, which is described by the corresponding values within the skinCoordWeight field of the Joint node. The skinCoordWeight field contains a list of floating point values, which describe an amount of "weighting" that should be used to affect the appropriate indices (as indicated by the skinCoordIndex field) of the skinCoord field of the Humanoid node. It is worth pointing out that the skinCoordWeight and skinCoordIndex fields are only used with continuous mesh models.

The Joint node is also used to store other joint-specific information. In particular, a joint name is provided so that applications can identify each Joint node at runtime. The Joint node may contain hints for inverse-kinematics systems that wish to control the H-Anim figure. These hints include the upper and lower joint limits, the orientation of the joint limits, and a stiffness/resistance value. Note that these limits are not enforced by any mechanism within the scene graph of the humanoid, and are provided for information purposes only. Use of this information and enforcement of the joint limits is up to the application.

Figure 10 describes the interface of the Joint node and the fields which allow the functionality of the node to be defined and accessed.

```
PROTO Joint [
   exposedField    SFVec3f     center           0 0 0
   exposedField    MFNode      children          [ ]
   exposedField    MFFloat     llimit           [ ]
   exposedField    SFRotation  limitOrientation   0 0 1 0
   exposedField    SFString    name             ""
   exposedField    SFRotation  rotation          0 0 1 0
   exposedField    SFVec3f     scale            1 1 1
   exposedField    SFRotation  scaleOrientation   0 0 1 0
   exposedField    MFInt32     skinCoordIndex     [ ]
   exposedField    MFFloat     skinCoordWeight    [ ]
   exposedField    MFFloat     stiffness         [ 0 0 0 ]
   exposedField    SFVec3f     translation        0 0 0
   exposedField    MFFloat     ulimit            [ ]
]
```

Figure 13. Joint Node Prototype

The center field specifies a translation offset from the root of the overall humanoid body description and is not intended to receive events in most cases. Since all of the Joint nodes have their center values defined in the same coordinate frame, the length of any segment can be determined by calculating the distance between the parent's Joint center and the child's Joint center. The exception will be segments at the ends of the fingers and toes, for which the end-effector Site nodes within the Segment node must be used.

The rotation field specifies a rotation of the coordinate system of the Joint node. fields necessary

The skinCoordWeight field contains a list of floating point values, which describe an amount of "weighting" that should be used to affect a particular vertex from the skinCoord field of the Humanoid node. Each item in this list has a corresponding index value in the skinCoordIndex field of the Joint node, which indicates exactly which coordinate is to be influenced.

The ulimit and llimit fields of the Joint PROTO specify the upper and lower joint rotation limits. Both fields are three-element MFFloats containing separate values for the X, Y and Z rotation limits. The ulimit field stores the upper (i.e. maximum) values for rotation around the X, Y and Z axes. The llimit field stores the lower (i.e. minimum) values for rotation around those axes. Note that the default values for each of these fields is [], which means that the joint is assumed to be unconstrained.

The limitOrientation exposedField gives the orientation of the coordinate frame in which the ulimit and llimit values are to be interpreted. The limitOrientation describes the orientation of a local coordinate frame, relative to the Joint center position described by the center exposedField.

The stiffness exposedField, if present, contains values ranging between 0.0 and 1.0 which give the inverse kinematics system hints about the "willingness" of a joint to move in a particular degree of freedom. For example, a Joint node's stiffness can be used in an arm joint chain to give preference to moving the left wrist and left elbow over moving the left shoulder, or it can be used within a single Joint node with multiple degrees of freedom to give preference to individual degrees of freedom. The larger the stiffness value, the more the joint will resist movement in the corresponding axis (X, Y, or Z for entries 0, 1 and 2 of the stiffness MFFloat field).

The name field is used for identifying the joints at runtime. Each Joint node should have a DEF name that matches the name field for that Joint, but with a distinguishing prefix in front of it. That prefix can be anything, but must be the same for all the Joint nodes in a particular humanoid. The distinguishing prefix is useful in the case of static routing to the Joint nodes of multiple humanoids in the same file. If only a single humanoid is stored in a file, the prefix should be "hanim_" (for Humanoid Animation). For example, the left shoulder would have a DEF name of "hanim_l_shoulder". The DEF name is used for static routing, which would typically connect OrientationInterpolators in the humanoid file to the joints. The name field must be present, so that applications are able to identify the Joint node at runtime.

It will occasionally be useful for the person creating a humanoid to be able to add additional joints to the body. The body remains humanoid in form, and is still generally expected to have the basic joints. However, they may be thought of as a minimal set to which extensions may be added (such as a prehensile tail). See the section on Non-standard Joints and Segments. If necessary, some of the joints (such as the many vertebrae) may be omitted. See Appendix A for details.

3. Deformation Of H-Anim Body

Realistic representation of human body can only be achieved by deformable humanoids. H-Anim 2001 working group constructed a mesh based deformation process for H-Anim bodies. Skin tag in the Humanoid node contains all the vertices those build human appearance. This provides integrity of the skin surface between the joints. By index based surface building process, skin coordinates may be read from skinCoord Index tag. Also skin coordinate normals are contained in the skinNormal tag. Each joint contains the indices of vertices in skinCoordIndex tag . This list of indices will be oriented by rotation transformations of joint they belong to and as well as the neighboring joints. SkinCoordWeight is the multiplicative value of an index. This value shows the percentage of the rotation, which will be applied to index

This design provides deformation engines all information they need. Following is a simple deformation engine process diagram.

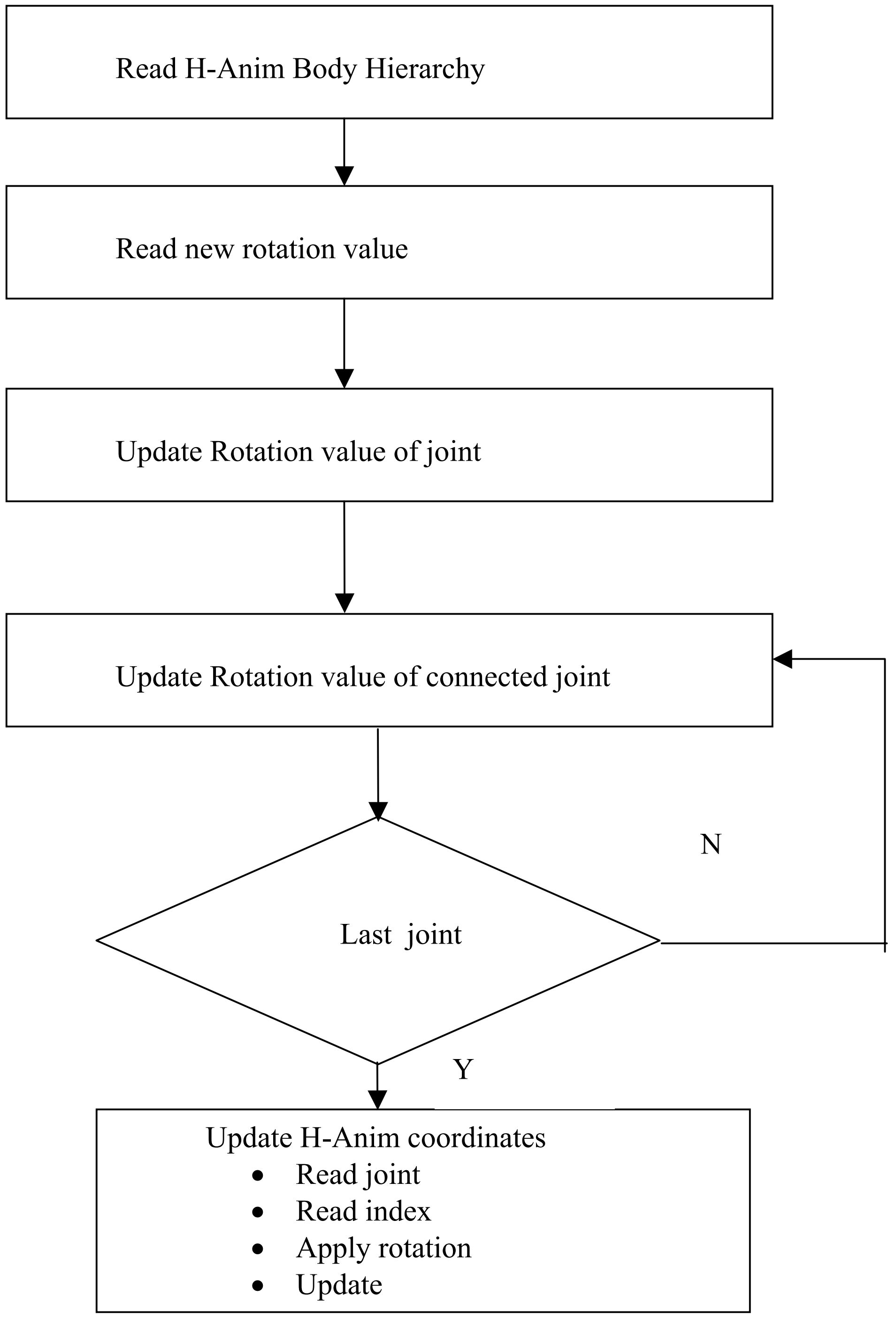

Read H-Anim Body Hierarchy
Read new rotation value
Update Rotation value of joint
Update Rotation value of connected joint
N
Last joint
Y
Update H-Anim coordinates
• Read joint
• Read index
• Apply rotation
• Update

Figure 14. Deformation Process

C. CONSTRUCTION OF VIRTUAL HUMAN FROM LASER SCAN

1. Method

In order to design a virtual Human, lasers scans from Cyberware© are used. This type of scan gives us more detailed skin information than the other scanning methods.[DUTTON2001].

a. Data Collection

Collection and ordering of the laser scan data is the primary task in designing a virtual human. Every scanning system has its own file format. This makes this work a little difficult. However, H-Anim 2001 standard uses a skin node, which collects all vertex information and constructs a surface by using indices from this field. By direct extraction of data cloud and putting it into this field provides an easy solution. (Figure 15)

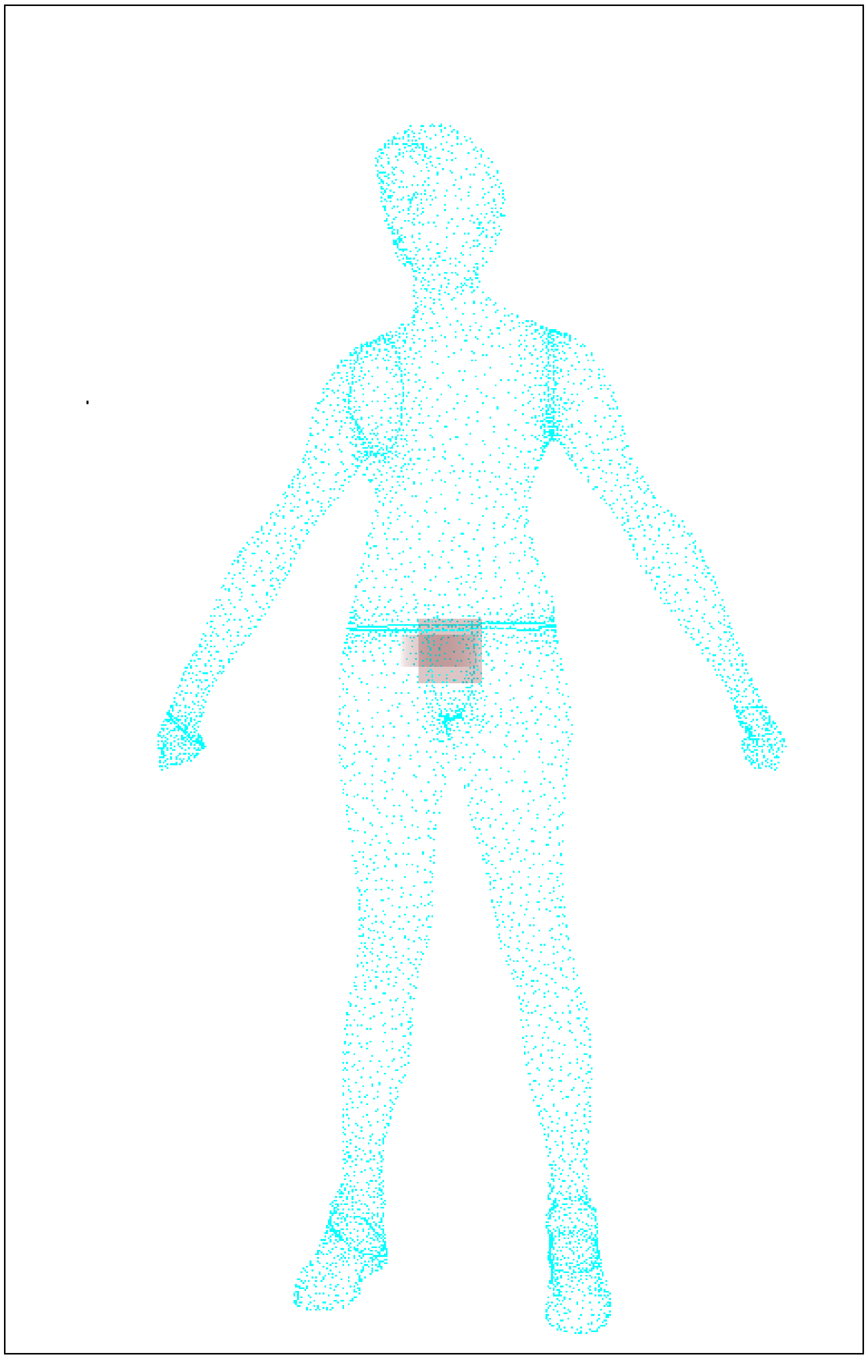


Figure 15. Laser Scan Data Cloud Courtesy of Cyberware©

Cyberware Laser scans include the polygon information for the scanned body skin. This information is used for mesh construction. This body mesh has a great number of triangles in order to provide a detailed appearance.(Figure 16) By using a triangle-reducing program, number of triangles can be adjusted for performance gain. In this research a free version of 3D reducer© from Sergei A Zavalesky is used. Triangle reduction must be made before segmentation process. (Figure 17) H-Anim standard does have **segment** node fields which are impossible to optimize with a standard tool.

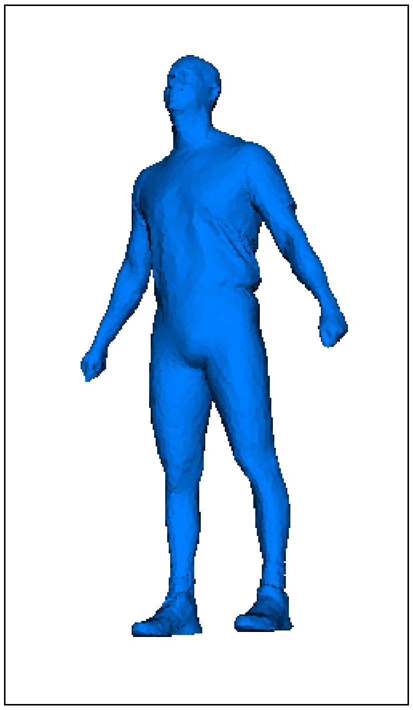


Figure 16. Laser Scan Polygon Mesh

b. Segmenting Process

For segmentation a simple bounded cylinder method is used. Segmenter program, written in Java and Virtual Reality Modeling Language (VRML), performs two tasks. First the program takes segment center coordinates for each body part as user input and finds all possible vertices inside a cylinder. Cylinder coordinates are the center of two segments, which are, connected each other. The user also defines end points of end effectors. The second task is writing the code in proper VRML format.

This method is an easy solution for segmentation process. Previous work [DUTTON2001] used a commercial product for this purpose. But this solution is expensive and time consuming. Because it is impossible to install that kind of professional modeling programs to everyone's computer and moreover modeling programs have a long learning curve. But modeling programs have high segmentation accuracy for polygonal mesh and more complex tools for rendering process. Accuracy near end points and around joints has a significant effect for deformation. This effect will be discussed in next section.

c. Deformation Issues

Simple Deformation Engine is a java thread, which works as a real-time rendering engine with VRML browser. Its performance is directly proportional to the number of vertices inside the humanoid mesh. If there are more then one humanoids inside the virtual environment then to increase performance we need to change level of detail during the construction of humanoid mesh.

Another issue is esthetic appearance of virtual human. Especially segmenting through false joint centers results abnormal articulation of body posture. This can be prevented by two methods. The first method is using a more accurate modeling tool. Because each person will have his own humanoid designed once. So someone can design it for you with an artistic ability and accuracy. The second solution is defining joints during scanning process. Which may be possible by putting a marker to the joints. Another way is standing with a special posture that shows all joint tearing points during scanning.

2. Results

Figure 17 shows the final product of the segmentation process. Different articulations are experimented with different orientation around x- axis. Figures 18 and 19 are arm and leg deformation examples. But simple deformation engine can not handle extreme orientation angles as seen Figure 20.

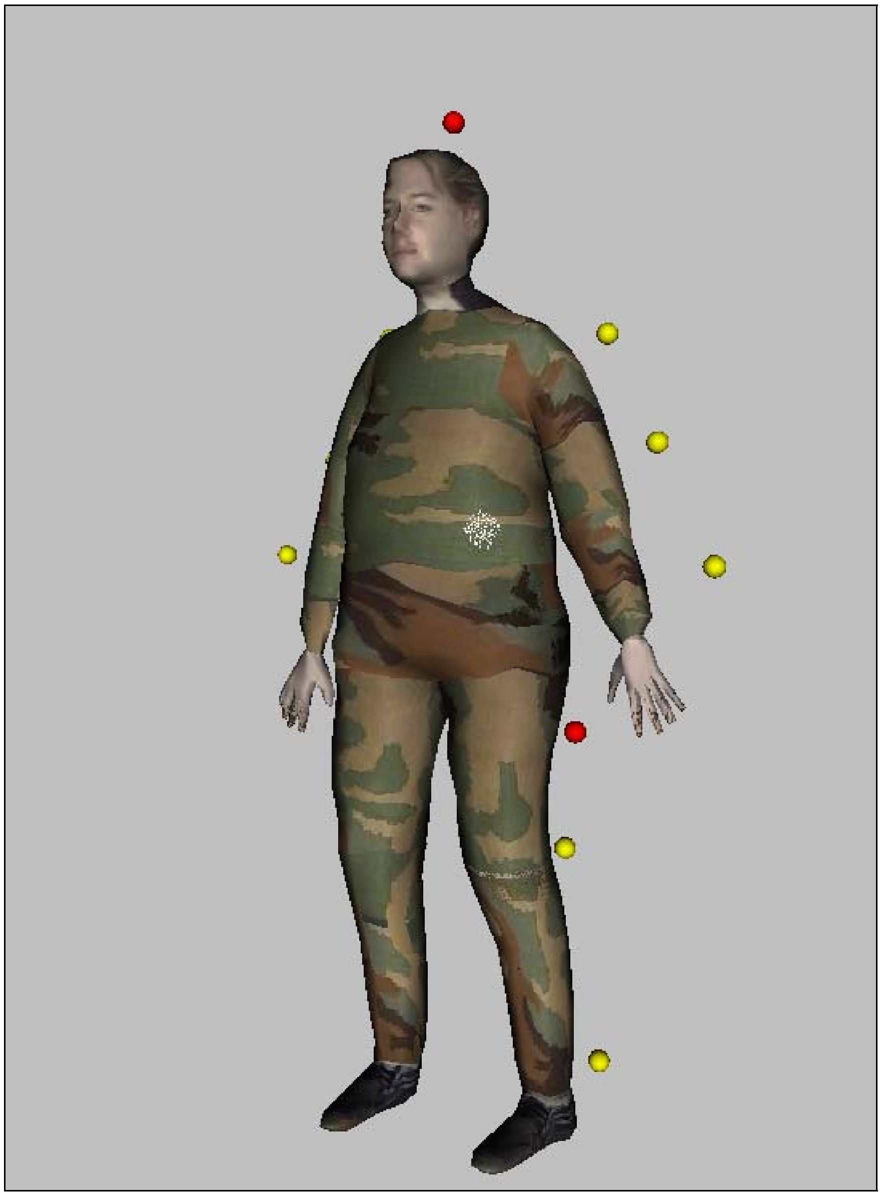


Figure 17. Final Product of Segmenter Software

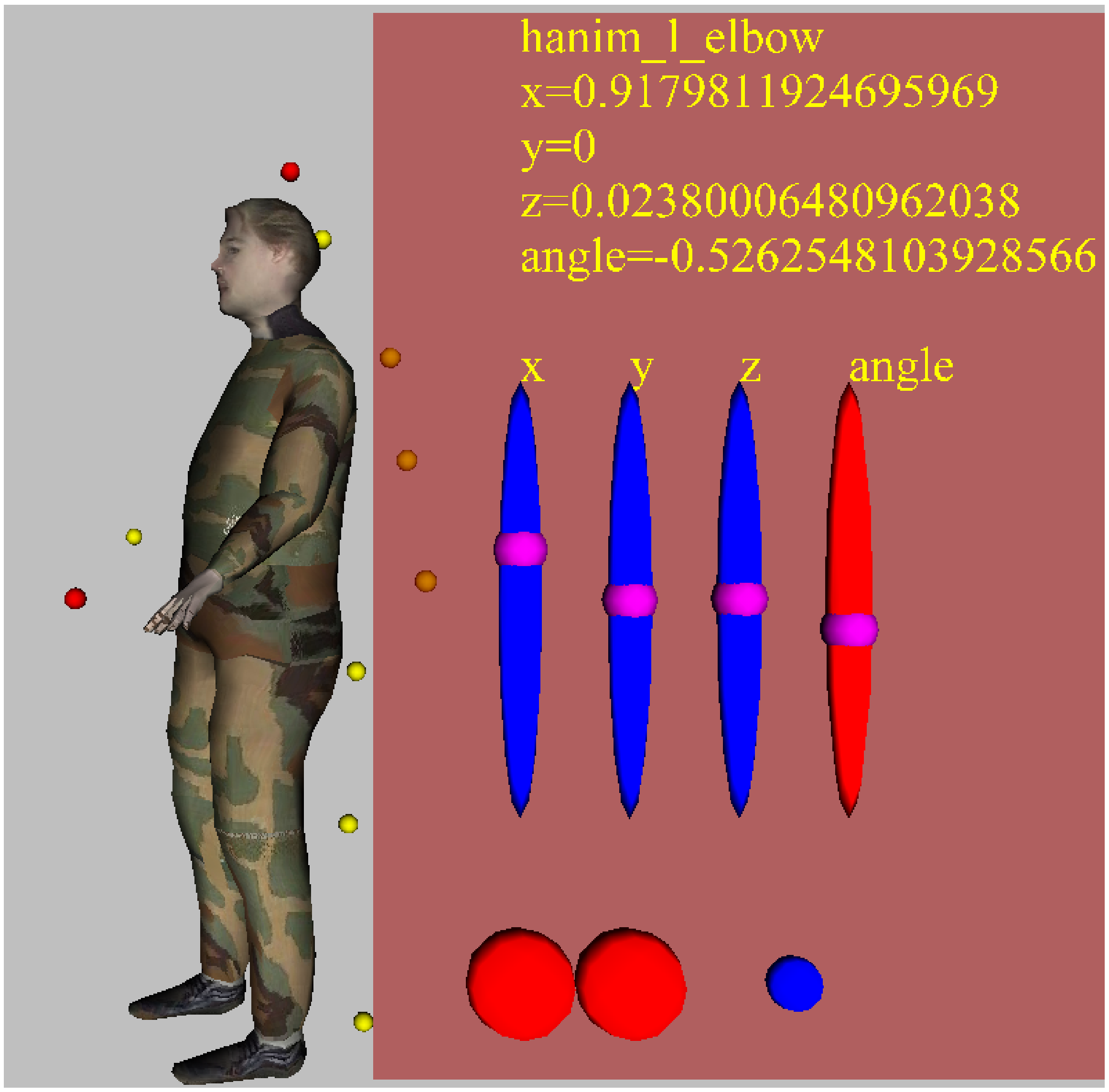

hanim_l_elbow
x=0.917981192469596
y=0
z=0.0238000648096203
angle=-0.526254810392856

Figure 18. Arm Deformation Example

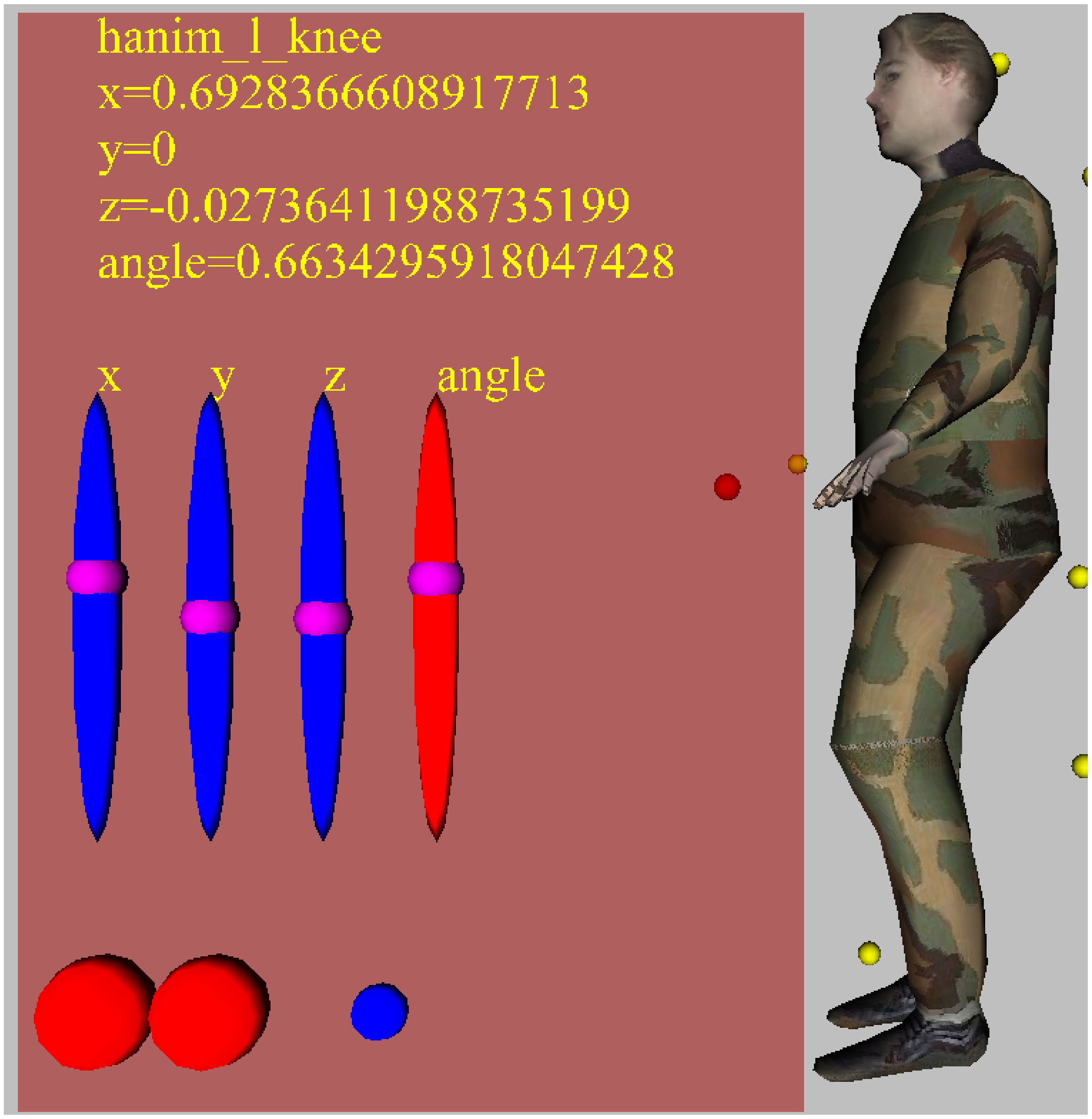

hanim_l_knee
x=0.6928366608917713
y=0
z=-0.02736411988735199
angle=0.6634295918047428

Figure 19. Leg Deformation Example

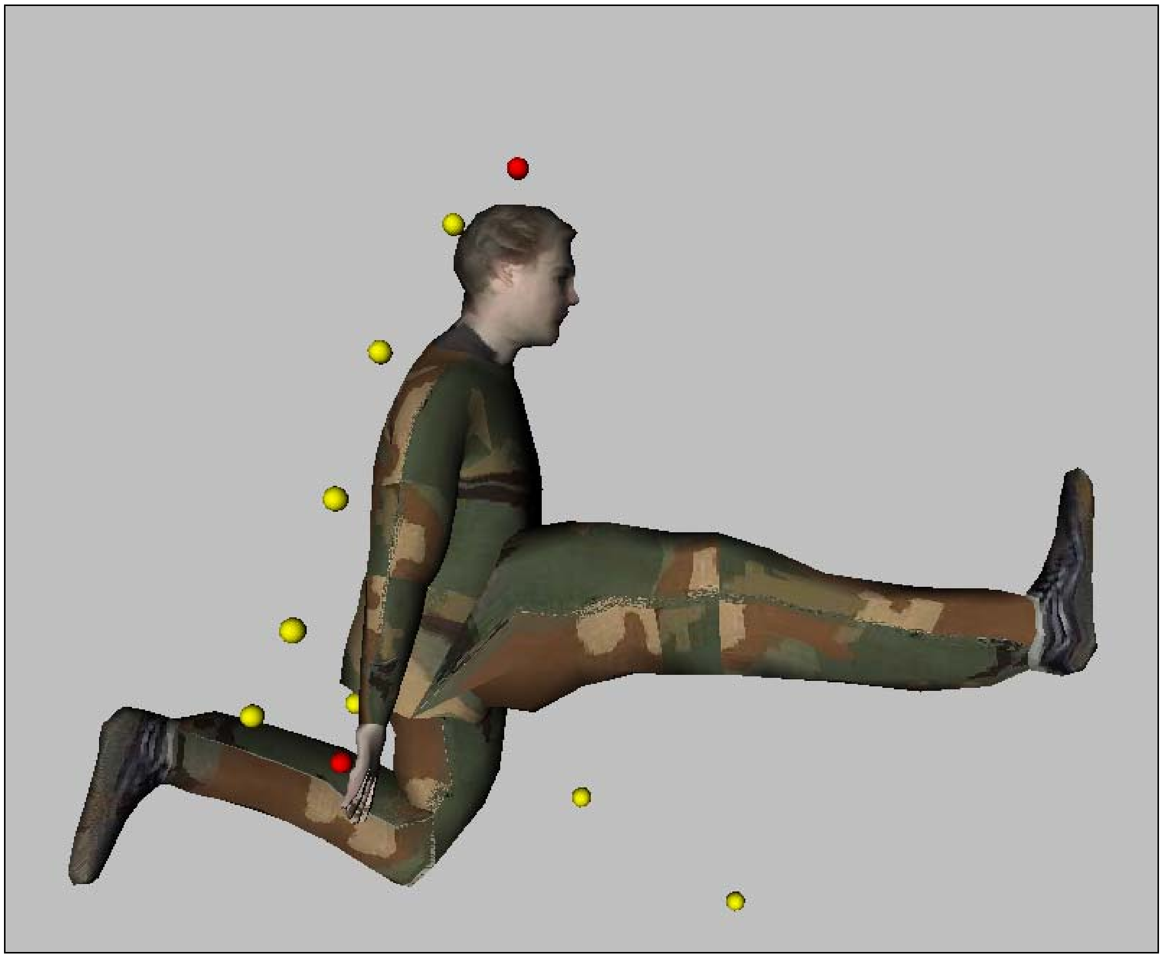


Figure 20. Deformation Problems

Human skin deformation is very complex process. Muscles under the skin have nonlinear stretching and expanding structure. Even though the joint node contains a weight field, which provides deformation coefficient for orientation, writing a complex deformation engine may decrease the performance. That's why we have static values for this field. And static values may create abnormal stretching polygons over the surfaces around the joints as seen in Figure 20.

IV. VIRTUAL HUMAN INTERFACE FOR MARG BODY TRACKING SYSTEM

A. ANALYSIS OF MARG HARDWARE

1. Orientation Signals

As stated before MARG system uses three-axis information from three sensors. Magnetic and accelerometer sensor give correction for drift and rate sensor give orientation information after using correction. All these nine signals are reduced to four tuples which are unit quaternion and defined as rotation around given vector. [BACHMANN2000] explains reduction of the X-matrix in detail. Optimized virtual human has fifteen segments as shown in Figure 21.

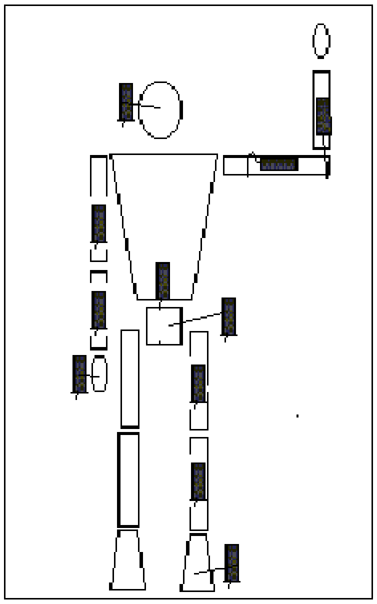


Figure 21. Segments for a virtual human in MARG

Currently there does not exist any graphics rendering hardware, which uses directly quaternion information. That is the reason why this system uses VRML language. VRML language rotations are defined as axis rotations, which is very similar to quaternion rotations. That's makes it easy to convert quaternion rotations to axis angle rotations. The drawback of this conversion is singularity of some angles.

New programmable vector graphics engines may provide a quaternion based calculation system. This approach will increase the performance of the system.

Our system uses quaternion orientation information over network. So network speed will directly affect frame rate. New algorithms, which may reckon articulations during loss of network information, may be developed as a new research project.

2. Location Signals

To locate a human body this research used GPS signals. Today there exist very small sized GPS instruments, which can directly be plugged to wearable computers. A parser class is written to read this information from a GPS by using NMEA standard 183. Location signals are taken as Latitude and Longitude and applied to the human root translation field. Because this field is the root of all joints and can translate all segments at the same time.

Error rates applied to GPS signals by government sometimes make it impossible to use them for human body location. That's why we used a differentiation method for localization. For this purpose two GPS instruments, one for tracker one for the actual person whose body is being tracked, are used. The difference between these two locations is taken and the location of the tracker is used as a reference to locate the actual tracked person. The error for the tracker and tracked person is same for the close regions. So relative position will be as accurate as actual location.(Figure 22) In other words $\mathbf{P_tP_r}$ will parallel to $\mathbf{P_{tgps}P_{rgps}}$. For the implementation this vector will be brought onto a special origin defined by the user.

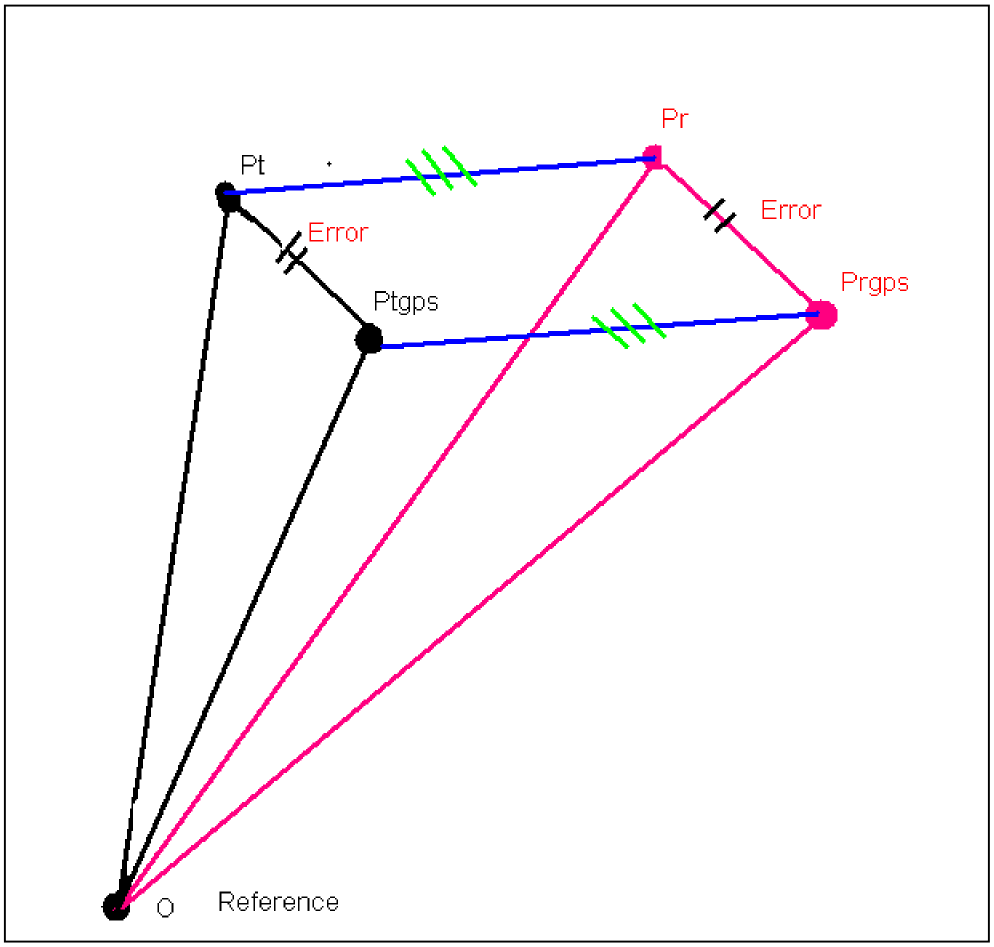


Figure 22. GPS Positioning by two GPS Instruments

B. IMPLEMENTATION AND RESULTS

Figure 23 is general architecture of the implemented system. Conceptually the system is designed at least for fifteen MARG sensors. But currently there exists only one sensor for experiments. However designing one channel is enough for the proof of concept because all other fourteen channels will have same architecture.

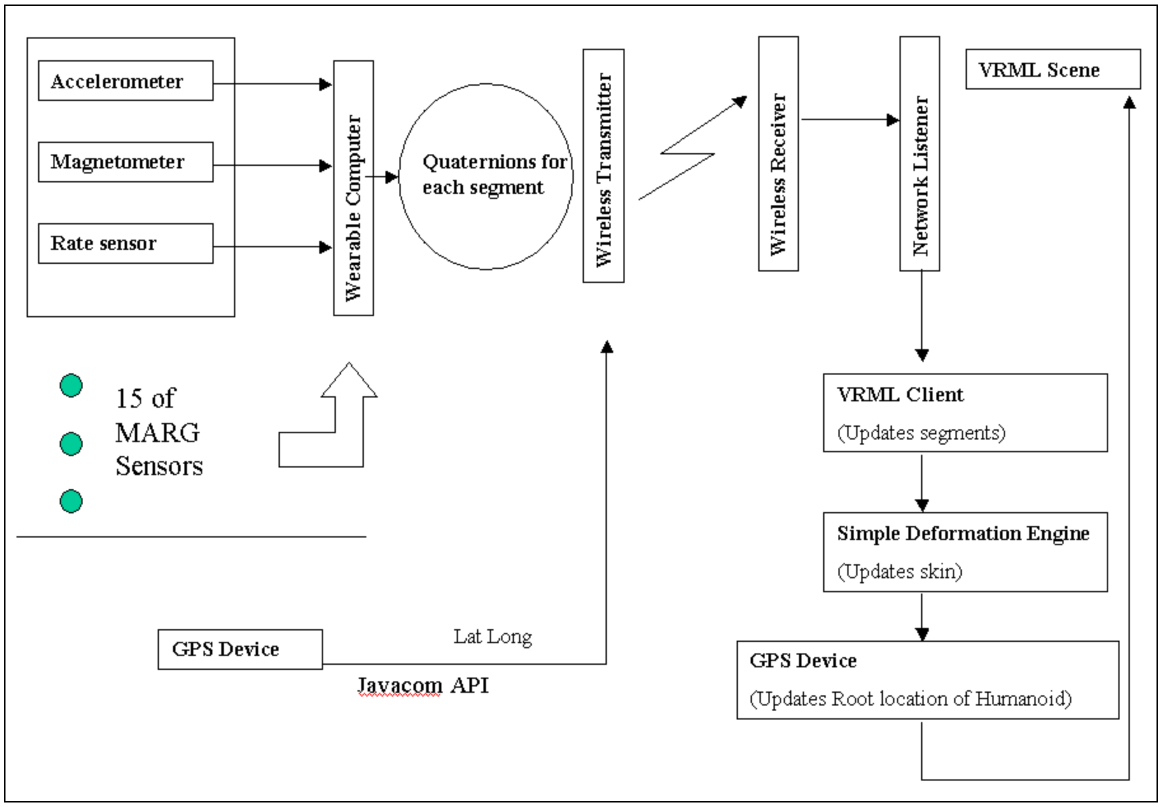


Figure 23. General Architecture

MARG Research group is working on wireless system, which is not implemented yet. Another part of the project deals with networking process. Because of these two ongoing researches this work implemented a simulated network server to read the quaternion orientation information. Simulated server sends previously recorded data through network to client software. Motion capture is done by MARG system designed by [BACHMANN2000].

Other function of Server is sending location information. Commercial GPS connected to serial port of computer updates position field of server. Special parser thread written in JAVA language reads communication port of computer, and decodes NMEA 183 standard data into text format. Finally the server program packs this data with quaternions.

There are two possible package types for this system. One of them is packing all rotation info into one package. Other possibility is sending this rotation info separately for each joint. Implemented system used second packet format as shown in Figure 24.

UDP HEADER	SEGMENT NAME	X Y Z W

Figure 24. Packet Format

Client program reads these network packages and updates the joint rotations and the humanoid root translation field. During this process simple deformation thread updates the skin in order to create seamless body form.

Performance of the system as frame per second is shown in Figure 24. A Pentium IV with GFORCE 2.5, 32 Mb graphics card has been used to make the experiments. Clearly level of detail is main reason for the decrease in performance.

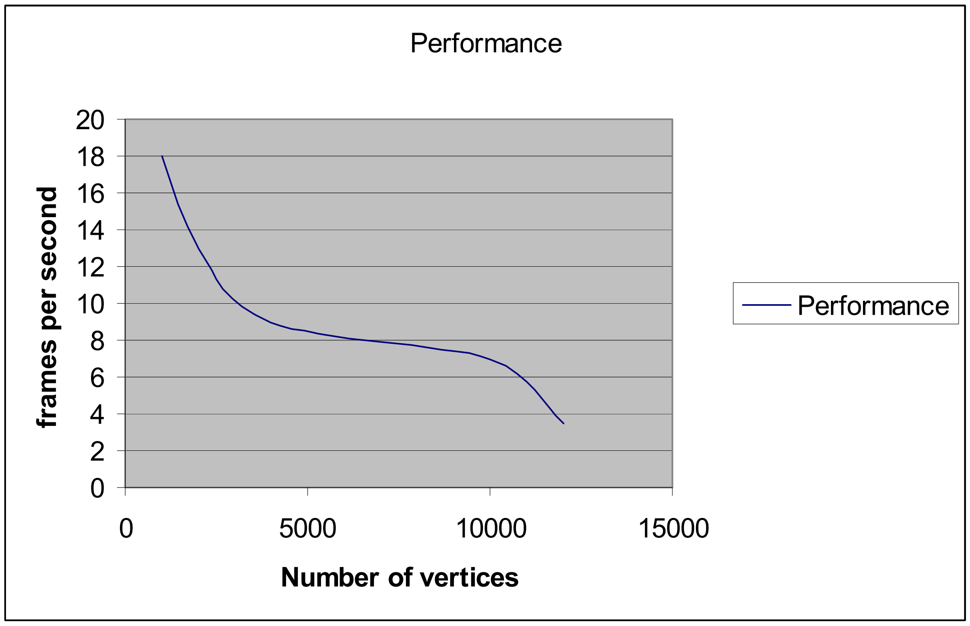


Figure 25. Performance Analysis

Another parameter effecting to the system performance is network delays. Network latency represents one of the biggest challenges for net-VE program writers.[ZYDA2000]. In order to decrease this effect server program sends user datagram packets (UDP) to the network. Disadvantage of the UDP is that they are not reliable. But number of packets sent for a human body is so many that loss of some packets will not directly affect the appearance.

Simple deformation engine is the most important and time-consuming thread for the rendering performance. Depending of the number of vertices this engine renders the surface data for each time a new rotation quaternion package updates the humanoid joint structure.

V. SUMMARY AND CONCLUSION

A. SUMMARY

This thesis presented a real-time method for the specification of seamless virtual human interface for MARG body tracking system based on H-Anim 2.0 standard. Implementation is realized by Java-VRML language.

Designing a virtual human has three aspects. First aspect is construction of a realistic graphical representation of human. This thesis used Cyberware© laser scans in order to achieve this goal. Second aspect is related to animation of this static structure. For this purposes, laser scan data cloud is segmented into a skeleton structure. This skeleton structure is a low level of detailed version of H-Anim 2.0 specification. H-Anim 2.0 standard also provided a seamless human skin structure.

Final aspect of virtual human design is to create a realistic and real-time articulation for each segment. Sourceless MARG sensor system, which is designed by Naval Postgraduate School researchers, is suitable motions capture system for this purpose. It uses singularity-free quaternions for rotation. Quaternions provide not only a safe mathematical base for rotations but also speed for the calculations of orientations.

Motion capture usually deals only with segment rotations. This research added position tracking to achieve the goal for a remote body tracking. Global positioning system is regarded as an optimal, cheap and global method to locate the human body.

B. CONCLUSIONS

Creating a realistic virtual world can be measured primarily by human perception. The more natural the objects are, the more comfortable becomes the human in this environment. But this also brings a problem. Humans become more selective in realistic virtual environment. As their expectation for reality increases, their forgiveness for any type of error decreases.

Keeping this in mind this research has following conclusions.

Realistic human body can be created by directly using laser scans and segmenting this scan by either a commercial 3-D modeling software or a simple segmentation VRML program.

Segmented human model will be able to get information for each segment through network by using MARG body tracking system.

Location information is usually regarded as secondary issue for body tracking. This research takes this issue as one of primary problems and implemented a solution by using GPS parser. But the error in GPS signals and lack of navigational position information in virtual worlds create drawback. Differential GPS systems can prevent this disadvantage by using different techniques.

Speed of Internet and increase of CPU power enabled ordinary users to be a part of 3D virtual worlds. Human existence in these worlds is currently possible by static avatars. Some of these avatars have limited animated behaviors for simple body movements like hand waving. Real-time sourceless body tracking will make it possible for these avatars to have infinitely many articulations with realistic skin deformation. This research proved this possibility by using MARG body tracking system which is, applied a VRML-H-Anim based avatar with GPS support.

LIST OF REFERENCES

BACHMANN99	Bachmann, E., Duman I., Usta, U., McGhee R., Yun, X., and Zyda, M., "Orientation tracking for Humans and Robots Using Inertial Sensors," *International Symposium on Computational Intelligence in Robotics & Automation* (CIRA 99), Monterey, CA, November 1999, pp.187-194.
BACHMANN2000	Bachmann Eric , *Inertial and MagneticTtracking of Limb Segment Orientation Inserting Humans Into Synthetic Environments,* Dissertation , Naval Postgraduate School, Monterey, CA, March 2000
CRAI89	Craig, J., *Introduction to Robotics: Mechanics and Control, Second Edition,* Addison-Wesley Publishing company, Inc., Menlo Park, CA, 1989.
COOKE92	Cooke, J., Zyda, M., Pratt, D., McGhee, R., "NPSNET: Flight Simulation Modeling Using Quaternions," *Presenc*e, Vol. 1, No. 4, MIT Press, Cambridge MA, Fall, 1992, pp. 404-420.
DUMAN99	Duman, I., *Design, Implementation and Testing of a Real-Time Software System for a Quaternion-Based Attitude Estimation Filter,* Master's Thesis, Naval Postgraduate School, Monterey, CA, March 1999.
DURL95	Durlach, N., and Mavor, A., National Research Council, *Virtual Reality: Scientific and Technological Challenge*s, National Academy Press, Washington, DC, 1995.
DUTTON2001	Dutton James Allen , *Developing Articulated Human Models From Laser Scan Data For Use As Avatars In Real-time Networked Virtual Environments* , Master's Thesis, Naval Postgraduate School, Monterey, CA, March 2001

EBERLY99	David Eberly *Quaternion Algebra and Calculus,*Magic Software 6006 Meadow Run Court Chapel Hill, NC 27516, eberly@magic-software.com
HANIM2001	H-Anim organization, *H-Anim 2.0 draft specification*, 2001 www.hanim.org
FRAN69	Frank, A. A., McGhee, R. B., *Some Considerations Relating to the Design of Autopilots for Legged Vehicle*s, *Journal of Terramechanic*s, Vol. 6, No.1, pp. 23-35, Pergamon Press, Great Britain, 1969.
FOLE97	Foley, J. D., Van Dam, A., Feiner, S. K. and Hughes J. F., *Computer Graphics Principles and Practice, Second Edition in* C, Addison-Wesley, 1997.
HAND93	Hand, C., "A Survey of 3-D Input Device*s*", Technical Report CS TR94/2, De Montfort University, Leicester, UK, 1993.
HAMILTON1899	Hamilton Sir William R. , *On quaternions* ,Royal Irish Academy press 1899
HODGINS2000	Hodgins Jesica, *Animating Human motion* Scientific American 1998 http://www.sciam.com/1998/0398issue/0398hodgins.html
MCGHEE93	McGhee, R. B., *CS4314 Class Notes: Derivation of Body Angular Rates to Euler Angle Rates Relationshi*p, Naval Postgraduate School, Monterey, CA, 1993.
MCGHEE97	McGhee, R. B., *CS4314 Class Notes: Finite and Infinitesimal Quaternion Rotation*s, Naval Postgraduate School, Monterey, CA, 1997.
MORSE53	Morse, Philip M.; Feshbach, Herman *Methods of Theoretical Physics* McGraw-Hill 1953 .

SHOEMAKE85 Shoemake, K., “Animating Rotation with Quaternion Curves”, *SIGGRAPH 8*5, pp. 245-254, ACM Press, 1985.

SHOEMAKE87 Ken Shoemake, *Animating rotation with quaternion calculu*s, ACM SIGGRAPH 1987, Course Notes 10, Computer Animation: 3–D Motion, Specification, and Control.

SKOP96 Skopowski, P., *Immersive Articulation of the Human Upper Body in a Virtual Environmen*t, Master’s Thesis, Naval Postgraduate School, Monterey, CA, December 1996.

THALMANN97 Daniel Thalmann , *Virtual Humans in Virtual Environments A New View of Multimedia Applications 1997* Computer Graphics Lab (LIG), Swiss Federal Institute of Technology (EPFL) CH-1015 Lausanne, Switzerland

USTA99 Usta, U., *Comparison of Quaternion and Euler Angle Methods for Joint Angle Animation of Human Figure Model*s, Master’s Thesis, Naval Postgraduate School,Monterey, CA, March 1999.

WATT98 Watt, A., and Watt, M., *Advanced Animation and Rendering Techniques, Theory and Practic*e, ACM Press, New York, NY, 1992.

WILKINS99 Wilkins , David R. *On Quaternions from Sir William Hamilton* Trinity College, Dublin 1999 http://www.maths.tcd.ie/pub/HistMath/People/Hamilton/

YUN99 Yun, X., Bachmann, E., McGhee R., Whalen, R., Roberts, R., Knapp, R., Healey, A., and Zyda M.,“Testing and Evaluation of an Integrated GPS/INS System for Small AUV Navigation,” *IEEE Journal of Ocean Engineerin*g, Vol. 24, No. 3, July 1999. pp. 396-404.

ZYDA99 Zyda M., and Singhal, S., *Networked Virtual Environments, Design and Implementatio*n, ACM Press, New York, NY, 1999.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Alper SINAV
   Turkish Navy
   Istanbul Turkey

4. Prof.Michael Zyda
   Naval Postgraduate School
   Monterey CA 93943-5101

5. Prof. Beny Neta
   Naval Postgraduate School
   Monterey CA 93940-5101

6. Prof. Xiaoping Yun
   Naval Postgraduate School
   Monterey CA 93940-5101

7. Deniz Kuvvetleri Komutanligi
   Ankara Turkiye